Método de autenticación seguro de usuarios de base de datos y de sistema para el desarrollo de aplicaciones web

Autor: Jonathan Franchesco Torres Baca

Lambayeque - Perú
2015

Agradecimientos

Gracias a la ciencia y a la tecnología, por una sociedad más culta e información libre.

Tabla de contenido

Tabla de Figuras

Introducción

Los datos de conectividad son de gran importancia para el desarrollo de software, estos permiten a los desarrolladores conectar sus aplicaciones a los servicios de almacenamiento, es común que estos datos sean confiados a los desarrolladores y es deber de ellos salvaguardar estos datos.

En la actualidad existen métodos de autenticación como oauth u openid que permiten autenticar ante un sistema usando un servicio o una cuenta, con el objetivo de no brindar la contraseña al sistema que deseamos autenticar, si bien estos métodos son de mucha utilidad, estos no están relacionas como el método de autenticación que se pretender crear, pues este método debe permiten en primer lugar salvaguardar los datos de conectividad al servicio de almacenamiento, en segundo lugar servir como método de autenticación de usuarios de sistema, en tercer lugar controlar y monitorear la actividad SQL de cada uno de los usuarios del sistema y por ultimo brindar un control sobre la conexiones al servicio de almacenamiento y el balanceo de carga de una aplicación.

La presente investigación mostrará como los datos de conectividad pueden ser robados, a través de métodos simples y otros a través de técnicas de ingeniería inversa, esto permitirá demostrar la falta de preocupación por salvaguardar los datos de conectividad por parte de los desarrolladores al momento de crear aplicaciones.

Justamente la principal razón de la investigación es crear un método independiente del lenguaje de programación y el servicio de almacenamiento que ayude a los desarrolladores a mantener seguros los datos de conectividad, pero a la vez ser usado como el método de autenticación de usuarios que permita monitorear la actividad SQL de cada uno de los usuarios conectados al sistema.

1. Generalidades.

1.1 Título:

Método de autenticación seguro de usuarios de base de datos y de sistema para el desarrollo de aplicaciones web.

1.2 Personal Investigador:

- ❑ Nombres y Apellidos: Jonathan Franchesco Torres Baca.
- ❑ Dirección: Av. Gran Chimú 1264, La Victoria, Chiclayo, Lambayeque.
- ❑ Teléfono: +51958429349
- ❑ Correo: chescot2302@gmail.com

1.3 Tipo de Investigación:

- ❑ Por su Finalidad: Básica.
- ❑ Por los Medios usados para obtener los datos: Experimental.
- ❑ Por el nivel de conocimiento que se adquiere: Explicativa.
- ❑ Por su campo de estudio: Científica.
- ❑ Por su razonamiento: Racional.
- ❑ Por los métodos usados:
    - ❑ Analítica.
    - ❑ Deductiva.

1.4 Área de Investigación: Tecnología.

1.5 Líneas de Investigación:

- ❑ Arquitectura de Software.
- ❑ Desarrollo de Software.
- ❑ Seguridad Informática.

1.6 Duración del Proyecto: 4 meses.

1.7 Fecha Inicio: 09/02/2015

1. 8 Fecha Fin: 24/05/2015

2. Aspectos de la Información.

2.1 Realidad Problemática.

2.1.1 Descripción Problemática:

En el desarrollo de software, la interacción con almacenes de datos es común al momento de desarrollar aplicaciones de cualquier índole, por esta razón, todo los desarrolladores buscamos documentación que ejemplifique la conectividad de aplicaciones con diferentes almacenes de datos, estos ejemplos nos enseñan a conectar nuestras aplicaciones usando varios métodos, los más comunes son guardando la información de conectividad en archivos de texto plano y otras embebiendo el código de conectividad al almacén de datos en el código fuente, siendo esta la más común, estas formas de conectividad vulneran la confidencialidad de los datos de conexión a los servidores que tienen instalado el servicio de almacenamiento, pues existe el riesgo que los usuarios finales o usuarios externos consigan los datos de conectividad fácilmente mediante la lectura del archivo de texto plano o a través de métodos especializados usando técnicas de ingeniería inversa y herramientas que permitan decompilar las aplicaciones para luego proceder con un proceso de análisis y así descubrir los datos de conectividad al servicio de almacenamiento.

La confidencialidad de los datos de conectividad al servicio de almacenamiento debe ser tratado con mucha cautela por los desarrolladores de software, pues se corre el riesgo que esta información sea divulgada y usada para tener acceso no autorizado al servicio de almacenamiento con propósitos poco éticos poniendo en riesgo la confidencialidad, integridad, consistencia, no repudio y disponibilidad de los datos.

Con el propósito de aclarar las ideas y entender la realidad problemática que se intenta dar solución, explicaremos que son datos de conectividad al servicio de almacenamiento y cuales son; además de desarrollar algunos casos que ejemplifican el problema.

¿Qué son los datos de conectividad al servicio de almacenamiento?

Son aquellos datos que permiten establecer o abrir un canal de comunicación entre nuestras aplicaciones y los distintos almacenes de datos (Oracle Database, Sql Server, Mysql, Postgres, etc.).

¿Cuáles son los datos de conectividad al servicio de almacenamiento?

Los datos de conectividad pueden variar de acuerdo al servicio de almacenamiento que se intenta conectar, pero sí sugerimos como ejemplo las bases de datos relacionales, los datos de conectividad serían los siguientes:

- Host (Equipo en donde se ejecuta el servicio de almacenamiento).
- Puerto (Puerta que permite la comunicación).
- Nombre de base de datos.
- Usuario de base de datos.
- Clave de base de datos.

Es clara la importancia de los datos de conectividad, por esta razón desarrollaremos algunos casos en las que se demuestra lo vulnerables que son.

Caso 1: Sistema Distributor.

Distributor es un sistema que se comercializa en todo el Perú, sobre todo para aquellos negocios de venta de equipos celulares y recargas virtuales, ha tenido como socio estratégico a Nextel(Entel), permitiendo que su sistema sea adquirido por varias empresas.

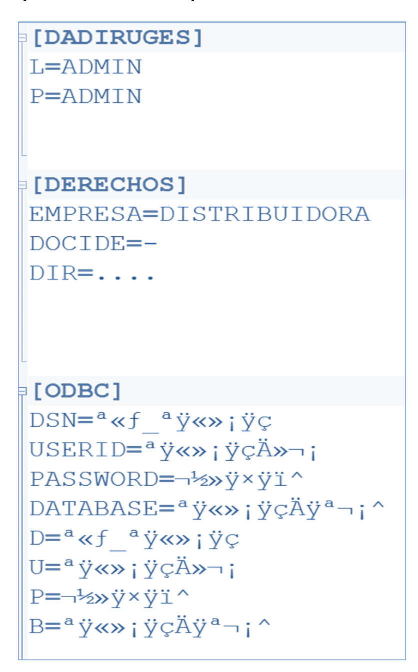
Especificaciones Técnicas:

- Lenguaje Visual Fox Pro.
- Base de datos Sql Server 2008
- Tipo de conexión ODBC.

El Problema:

Distributor es un sistema desktop orientado a resolver necesidades de empresas comerciales, desarrollado en Visual Fox Pro y base de datos SQL Server 2008. Si bien el sistema resuelve las necesidades del negocio, este no salvaguarda los datos de conectividad al sql server, ya que estos datos son expuestos en un archivo de texto plano, en cual los

datos son encriptados, esto puede notarse en la figura 1.



```
[DADIRUGES]
L=ADMIN
P=ADMIN

[DERECHOS]
EMPRESA=DISTRIBUIDORA
DOCIDE=-
DIR=....

[ODBC]
DSN=ª«ƒ_ªÿ«»¡ÿç
USERID=ªÿ«»¡ÿçÄ»¬¡
PASSWORD=¬½»ÿ×ÿï^
DATABASE=ªÿ«»¡ÿçÄÿª¬¡^
D=ª«ƒ_ªÿ«»¡ÿç
U=ªÿ«»¡ÿçÄ»¬¡
P=¬½»ÿ×ÿï^
B=ªÿ«»¡ÿçÄÿª¬¡^
```

Figura 1: Configuración ODBC Distributor

En la figura 2, notamos que también se usan archivos de texto plano para configuraciones del sistema y el dato más importante es el encuentra en el apartado ODBC, el cual permite saber cuál es el nombre del servidor que tiene alojado el servicio de almacenamiento.

```
[GENERAL]
ANHO=
SERIE_FACTURACION=001
SERIE_GUIAREMISION=001
SERIE_NOTACRED=1

[DERECHOS]
EMPRESA=DISTRIBUTOR
DOCIDE=
DIRECCION=
TELEFONO=

[ODBC]
SERVER=SERVIDOR001
DSN=
USERID=
PASSWORD=
DATABASE=

[APLICACION]
NOMBRE=Modulo de Comercialización
ABRV=Mod_Comercial
REGISTRADO=S
SISTEMA=0001
MODULO=0002
BITMAP=

[MONEDA]
SIMBOLO_E=US$
SIMBOLO_N=S/.
MONEDA_E=Dólares
MONEDA_N=Nuevos Soles
TIPO_CAMBIO=3.52

[STOCKS_PEDIDOS]
RESTRINGIR=S

[CARGA_CAMION]
PEDIDOS=999
CARGA_CAMION=12000

[IMPRESION_TICKETS]
LOCINFO1=
LOCINFO2=
LOCINFO3=
LOCINFO4=

[PLANOS]
LUPDATE_FECHA=2004, 2, 6, 9,31,36
```

Figura 2: Configuración de sistema distributor

Al encriptar los datos cumple el objetivo de no exponer los datos de conectividad abiertamente, pero su encriptación es vulnerable por el método ODBC usado para establecer la conectividad, el cual obliga a crear un ODBC en cada uno de los clientes en lo que se va a ejecutar el sistema distributor, permitiendo de esta manera revisar la configuración del ODBC y así establecer una relación entre el archivo de texto plano, con datos de conectividad encriptados y la configuración del ODBC en cada una de las máquinas cliente.

En la figura 3, figura 4 y figura 5 mostramos como acceder a la configuración del ODBC, primero accedemos al “Panel de control” del sistema operativo Windows, luego pulsamos en la opción “Herramientas Administrativas”, en la siguiente pantalla en Orígenes de datos ODBC.

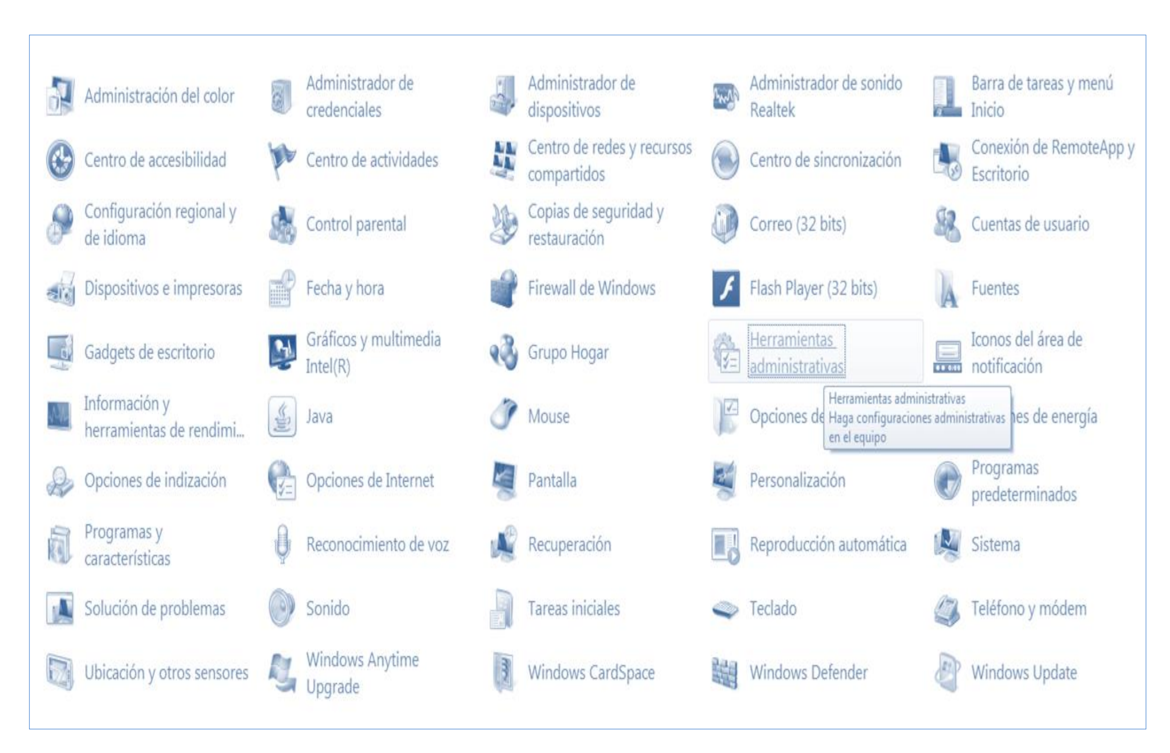


Figura 3: Panel de control



Figura 4: Herramientas administrativas

En la figura 5 observamos resaltado el nombre de origen ODBC "DSN_DISTRIB", el cual es usado por el sistema distributor para establecer conexión con el servicio de almacenamiento. Este primer valor permite establecer una conexión directa con el archivo de texto plano que contiene la configuración ODBC, específicamente la variable "DSN", con la cual podemos inferir que el sistema de encriptación es basada en diccionario, sugiriendo que las letras del abecedario sean reemplazados por caracteres especiales.



Figura 5: Administrar ODBC

Al editar la configuración ODBC "DSN_DISTRIB" observamos una pantalla como la de la figura 6, en donde podemos obtener el usuario de base de datos que permite la conexión o abrir sesiones con el servidor de base de datos SQL SERVER.

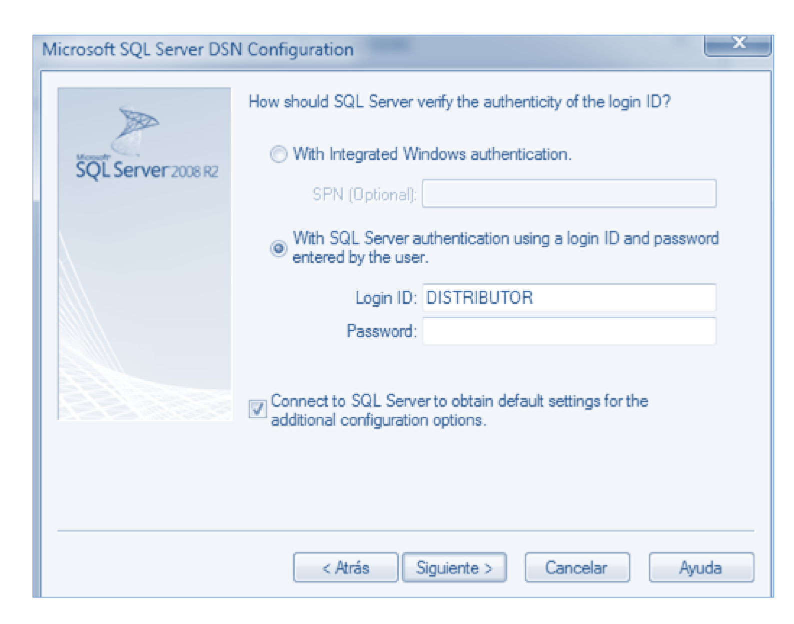


Figura 6: Configuración DSN

Con los datos obtenidos en la configuración del ODBC facilita la deducción de los valores encriptados en el archivo de texto plano de la configuración del distributor, en la figura 7 se presenta la resolución de los valores.

```
[DADIRUGES]
L=ADMIN
P=ADMIN
[DERECHOS]
EMPRESA=DISTRIBUIDORA
DOCIDE=-
DIR=....
[ODBC]
DSN=ª«ƒ_ªÿ«»¿ÿç   (11)
ª=D
«=S
ƒ=N
_=_
ª=D
ÿ=I
«=S
»=T
¿=R
ÿ=I
ç=B
USERID=ªÿ«»¡ÿçÄ»¬¿ (11)
ª=D
ÿ=I
«=S
»=T
¿=R
ÿ=I
ç=B
Ä=U*
»=T
¬=O*
¿=R
PASSWORD=¬¾»ÿ×ÿ¿^    (8)
DATABASE= (13)
ª=D
ÿ=I
«=S
»=T
¿=R
ÿ=I
ç=B
Ä=U
ÿ=I
ª=D
¬=O
¿=R
^=A
```

Figura 7: Resolución de datos de conectividad distributor

Caso 2: Sistema Retencion_Olano.

Retencion_Olano es un sistema de registro de retenciones del grupo olano.

Especificaciones Técnicas:

- Lenguaje Visual Net.
- Base de datos Sql Server 2008

El Problema:

Retencion_Olano es un sistema Desktop con acceso a datos en SQL Server 2008, este sistema tiene embebido en el código fuente los datos de conectividad al servicio de almacenamiento, los cuales obtendremos a través del decompilador net "JetBrains dotPeek 1.2", en la figura 8 observamos tanto el instalador de dotPeek cómo el archivo .EXE del sistema.



Figura 8: Sistema retencion_olano

En la figura 9 se muestra el sistema Retencion_Olano decompilado, en el cual podemos encontrar la clase que contiene los datos de conectividad al servicio de almacenamiento, demostrando así que tener estos datos embebidos no es fiable.

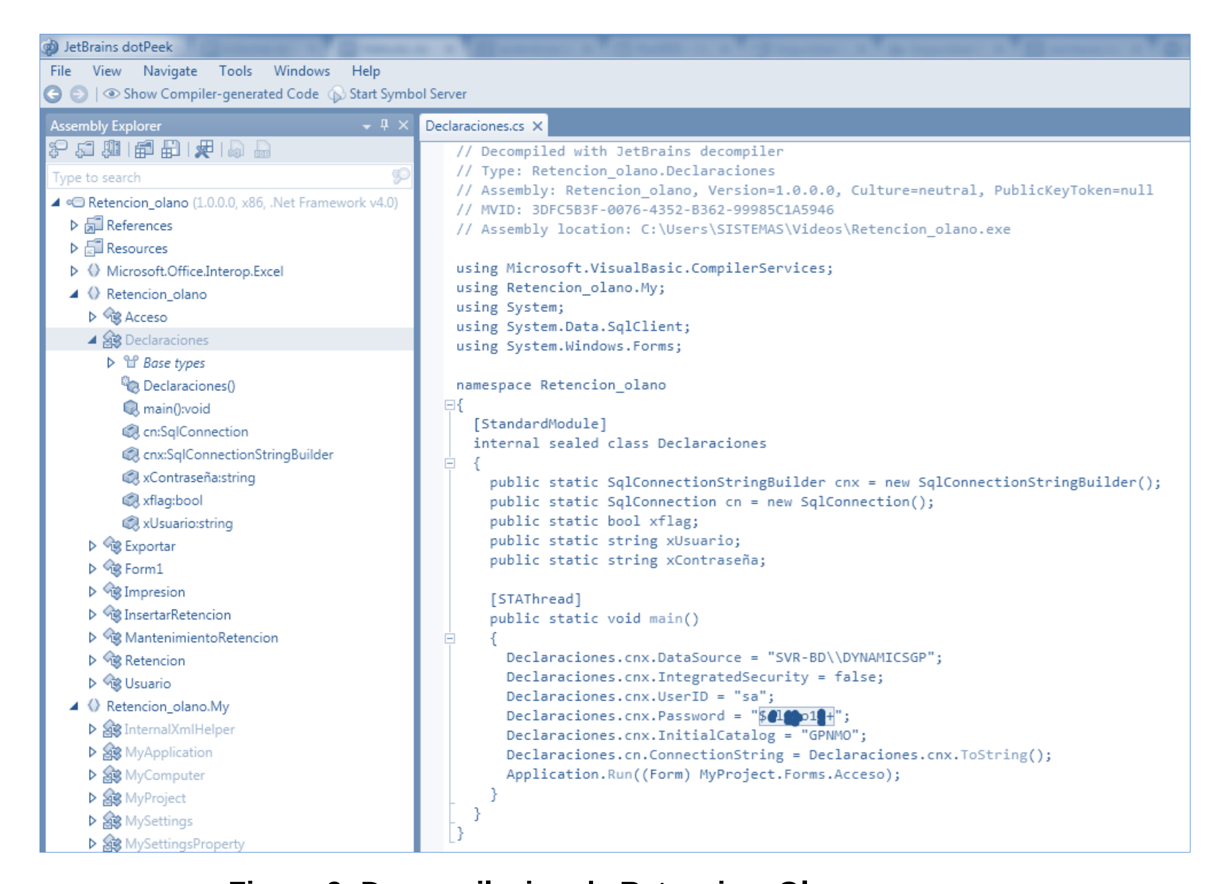


Figura 9: Decompilacion de Retencion_Olano

Caso 3: Sistema GPS.

GPS es un sistema de registro de proyecciones sociales realizadas por estudiantes universitarios de una determinada universidad.

Especificaciones Técnicas:

- Lenguaje Java.
- Base de datos Oracle 11g r2.
- Tipo conexión JDBC.

El Problema:

GPS es un sistema Desktop con acceso a datos en Oracle 11g r2, este sistema tiene embebido en el código fuente los datos de conectividad al servicio de almacenamiento, los cuales se obtendrán a través del decompilador Java “Java Decompiler”, en la figura 10 se observa tanto el archivo Jar del decompilador cómo del sistema.

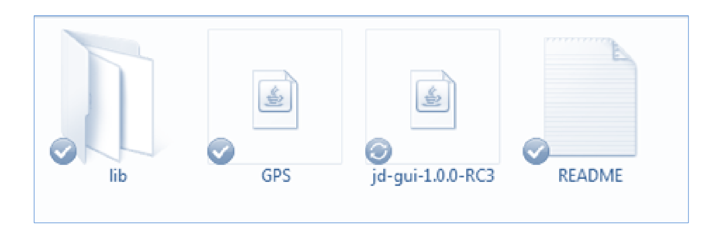


Figura 10: Sistema GPS

En la figura 11 se observa el archivo GPS.jar decompilado, este nos muestra toda la estructura de paquetes y clases. La clase de interés es aquella que contiene los datos de conectividad al servicio de almacenamiento, reforzando así la teoría de la vulnerabilidad de estos datos embebidos en el código fuente.

```
package com.gps.dao;

import com.gps.view.ViewApplication;
import java.io.PrintStream;
import javax.swing.JLabel;

public class ObjetoConexion
{
  private Conexion oracle;

  public Conexion conectarORACLE()
  {
    try
    {
      this.oracle = new Conexion("127.0.0.1", "1521", "orcl", "gpsadmin", "admin", "ORACLE");
      boolean estado = this.oracle.conectarBD();
      if (estado)
      {
        System.out.println("CONECTADO CORRECTAMENTE");
        return this.oracle;
      }
      System.out.println("ERROR AL CONECTAR");
      return null;
    }
    catch (Exception ex)
    {
      System.out.println("ERROR AL CONECTAR: " + ex.getCause());
      ViewApplication.mensajeEstado.setText("Error al conectar a la base de datos");
    }
    return null;
  }

  public Conexion getOracle()
  {
    return this.oracle;
  }
}
```

Figura 11: Decompilacion de Sistema GPS

Caso 4: Sistema Navasoft.

Navasoft es un sistema comercial que ha tenido gran demanda en la región Lambayeque.

Especificaciones Técnicas:

- Lenguaje Visual Fox Pro 9.0.
- Base de datos SQL Server 2008 r2.
- Tipo de conexión ODBC

El Problema:

Navasoft es un sistema Desktop con acceso a datos SQL Server 2008, este sistema tiene los datos de conectividad al servicio de almacenamiento en archivo de texto plano, los cuales podemos verificar en la figura 12.

```
(Encabezado)

SERVER      = SVR-BD\DYNAMICSGP
PWD         = $[redacted]1[redacted]1[redacted]+
UID         = sa
DRIVER      = {SQL server}
LENGUAJE    = Español
CONNECT     = 0
VALCOSTO    =          17
QUERYTIMEOUT = 0
SMTPCONNEC= 1
UPDATE = ON
```

Figura 12: Configuración ODBC Navasoft

Caso 5: Sistema Tramite.

Tramite es un sistema de trámite documentario desarrollado como ejemplo de programación.

Especificaciones Técnicas:

- Lenguaje Java.
- Base de datos Mysql.
- Tipo de conexión JDBC.

El Problema:

Tramite es un sistema web desarrollado en java como ejemplo de en programación, este ejemplo permitirá reforzar la problemática que afirma la vulnerabilidad de los datos de conexión al servicio de almacenamiento. En las figura 13 se observa la estructura de un proyecto web, en esta estructura se puede encontrar la carpeta "dist" generada a partir de un Build del IDE que usemos.

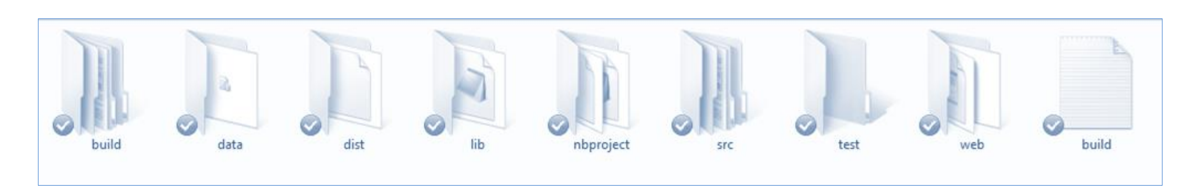


Figura 13: Sistema de Tramite documentario

En la Figura 14 se observa en el archivo "Tramite.war", este archivo permite el encapsulado del desarrollo de una aplicación web, permitiendo el deployment de estas aplicaciones en un servidor web o contenedores de servlets.



Figura 14: War Tramite

En las Figura 15, Figura 16, Figura 17 y Figura 18 se puede observar el proceso en que se descomprime el archivo “Tramite.war” el archivo war tiene el mismo comportamiento que un archivo Zip. Al descomprimir este archivo se puede navegar por la estructura de carpetas y buscar los archivos Class a decompilar y así obtener los accesos a los datos de conectividad

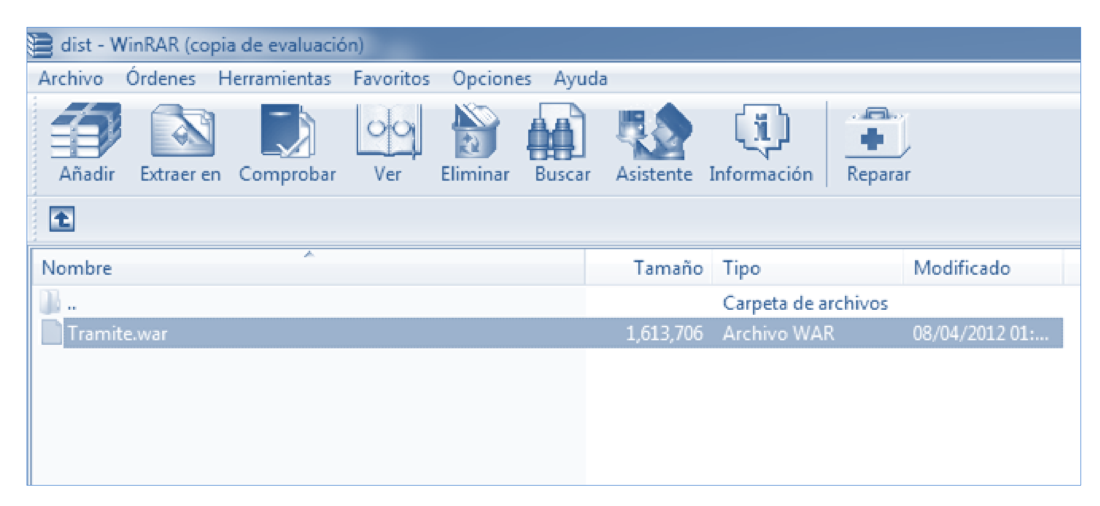


Figura 15: Winrar - Tramite

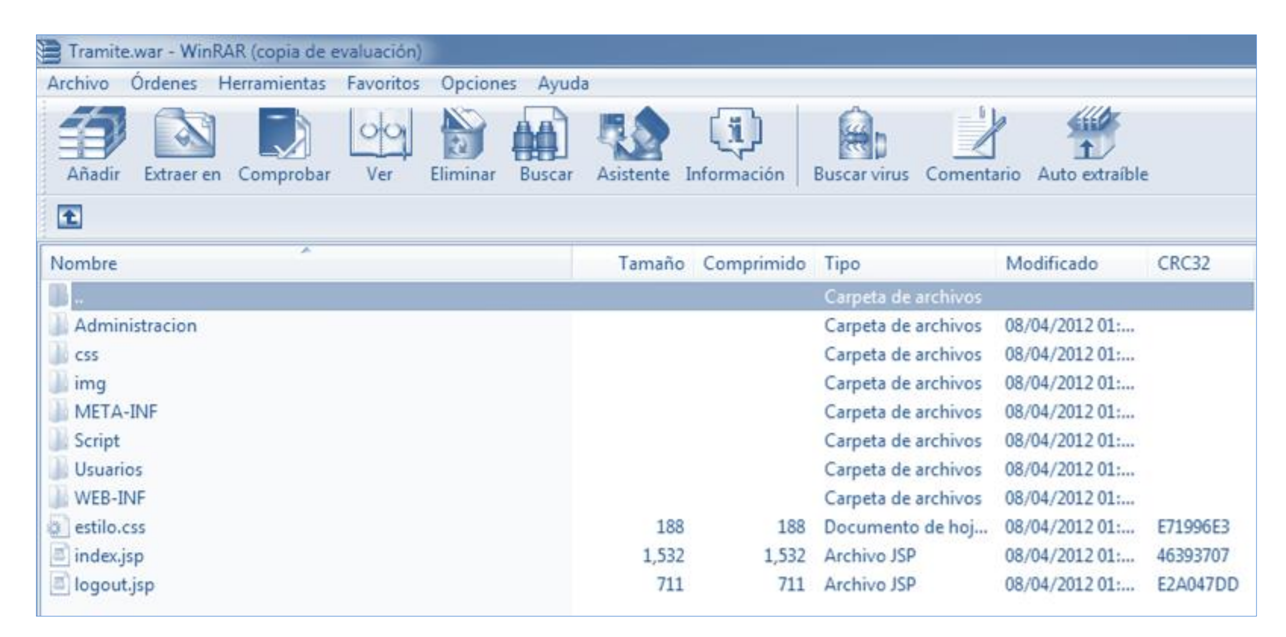


Figura 16: Estructura de directorio de War tramite

Tramite.war - WinRAR (copia de evaluación)

Archivo Órdenes Herramientas Favoritos Opciones Ayuda

Añadir Extraer en Comprobar Ver Eliminar Buscar Asistente Información Buscar virus Comentario Auto extraíble

Nombre	Tamaño	Comprimido	Tipo	Modificado	CRC32
..			Carpeta de archivos		
Administracion			Carpeta de archivos	08/04/2012 01:...	
css			Carpeta de archivos	08/04/2012 01:...	
img			Carpeta de archivos	08/04/2012 01:...	
META-INF			Carpeta de archivos	08/04/2012 01:...	
Script					
Usuarios					
WEB-INF					
estilo.css	188				
index.jsp	1,532				
logout.jsp	711				

Carpeta y opciones de extracción

General Avanzado

Carpeta de destino (si no existe, se creará)

C:\Users\SISTEMAS\Desktop

Mostrar

Nueva carpeta

Modo de actualización

Extraer y reemplazar

Extraer y actualizar

Solo actualizar

Modo sobrescribir

Confirmar sobreescritura

Sobrescribir sin confirmación

Omitir ficheros existentes

Renombrar automáticamente

Varios

Extraer en subcarpetas

Conservar ficheros dañados

Mostrar ficheros en el Explorador

Guardar opciones

Escritorio

Bibliotecas

SISTEMAS

Equipo

Disco local (C:)

Disco local (D:)

Disco local (E:)

Disco extraíble (G:)

Red

DISTRIBUTOR

Escritorio

Aceptar Cancelar Ayuda

Figura 17: Extraer Tramite.war

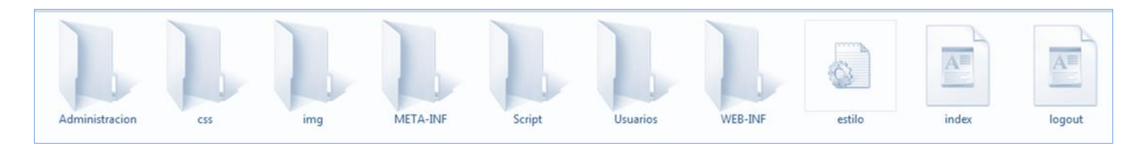


Figura 18: Estructura extraida de Tramite.war

En la figura 19 se observa que el directorio que surgió después de descomprimir el archivo “Tramite.war”, este lleva el mismo nombre y que los datos de conectividad al servicio de almacenamiento se encuentra en el archivo class “ObjetoConexion”, esta clase será decompilada y se mostrará los datos de conectividad tal cual se puede observar en la figura 20, dejando en claro que no solo en los sistemas desktop son vulnerables los datos de conectividad.

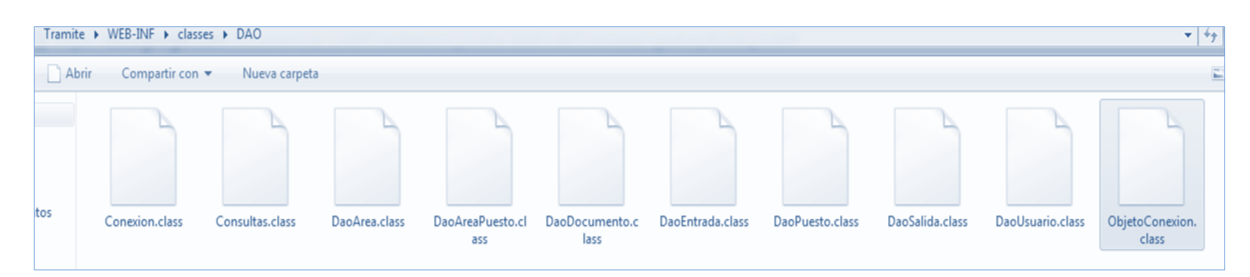


Figura 19: Archivos class de capa DAO

Después de haber analizado los siguientes casos, se puede afirmar que los datos de conectividad son vulnerables.

ObjetoConexion.class - Java Decompiler

File Edit Navigation Search Help

classes

- BEAN
- DAO
  - Conexion.class
  - Consultas.class
  - DaoArea.class
  - DaoAreaPuesto.class
  - DaoDocumento.class
  - DaoEntrada.class
  - DaoPuesto.class
  - DaoSalida.class
  - DaoUsuario.class
  - ObjetoConexion.class
- LOGIC
- SERVLETS

ObjetoConexion.class

```java
package DAO;

import java.io.PrintStream;
import java.sql.Connection;

public class ObjetoConexion
{
  private static Conexion sql;
  private static Conexion mysql;
  private static Conexion oracle;

  public static Conexion getMysql()
  {
    return mysql;
  }

  public static Conexion getOracle()
  {
    return oracle;
  }

  public static Conexion getSql()
  {
    return sql;
  }

  public static Connection conectarSQL()
  {
    try
    {
      sql = new Conexion("127.0.0.1", "1433", "dbia", "sa", "", "SQL");
      boolean estado = sql.conectarBD();
      if (estado)
      {
        System.out.println("CONECTADO CORRECTAMENTE");
        return sql.getCnx();
      }
      System.out.println("ERROR AL CONECTAR");
      return null;
    }
    catch (Exception ex)
    {
      System.out.println("ERROR AL CONECTAR: " + ex.getCause());
    }
    return null;
  }

  public static Connection conectarMYSQL()
  {
    try
    {
      mysql = new Conexion("127.0.0.1", "3306", "tramite", "root", "toor", "MYSQL");
      boolean estado = mysql.conectarBD();
      if (estado)
      {
        System.out.println("CONECTADO CORRECTAMENTE");
        return mysql.getCnx();
      }
```

Figura 20: Decompilacion de archivo class de conectividad a bd

2.1.2 Formulación del Problema:

¿Cómo mantener la confidencialidad de los datos de conectividad al servicio de almacenamiento para el desarrollo de software?

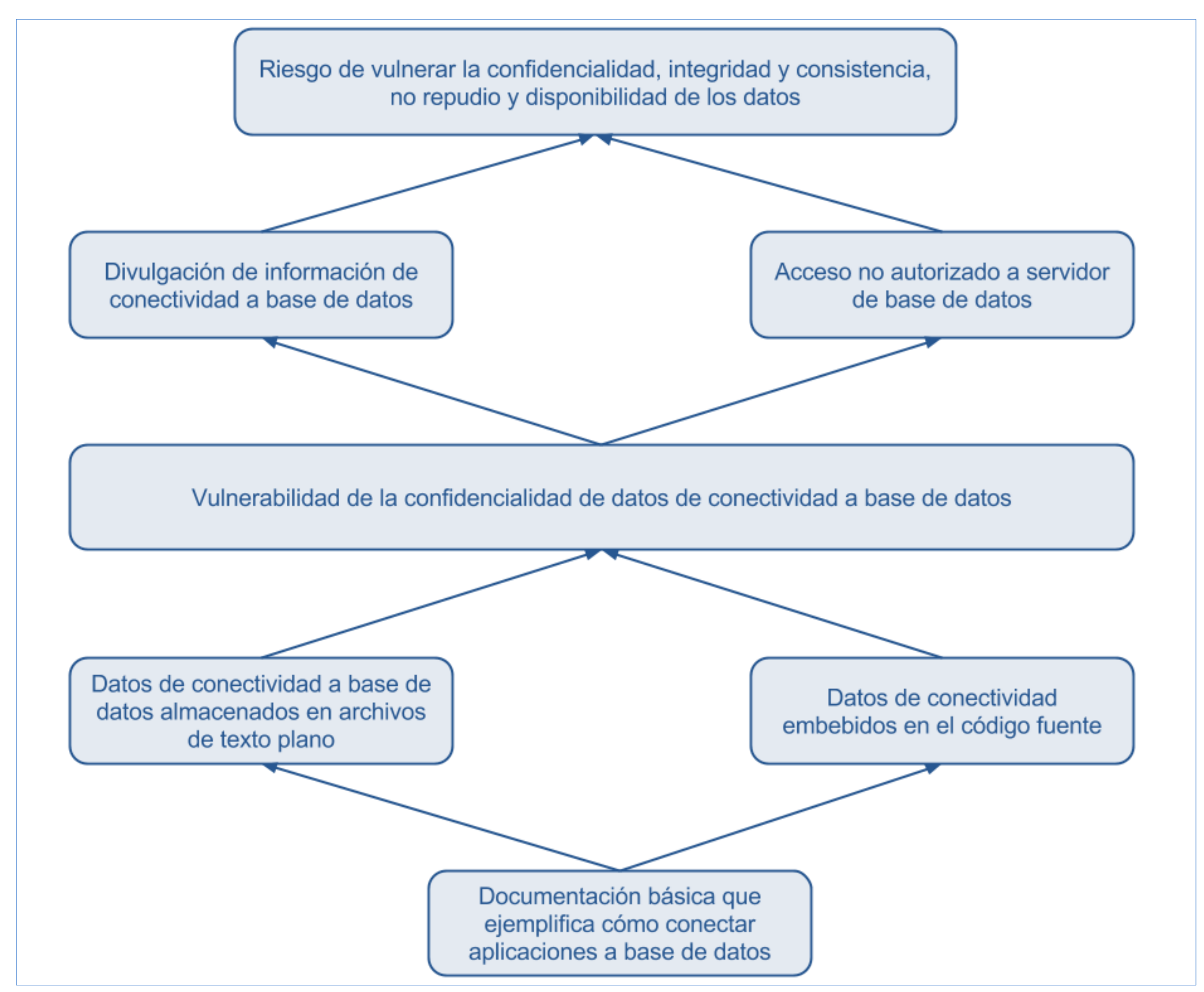


Figura 21: Árbol de problema

2.1.3 Justificación e importancia del estudio:

La confidencialidad de los datos de conectividad al servicio de almacenamiento al momento de desarrollar aplicaciones es muy importante, y es el deber de todos los desarrolladores disminuir el riesgo haciendo uso de lenguajes compilados, ofuscadores y más. En este proyecto se pretende desarrollar un método que permita asegurar la confidencialidad de los datos de conexión, además de usarlo como método de autenticación de usuarios de sistemas multiempresa.

2.1.4 Objetivos del Estudio.

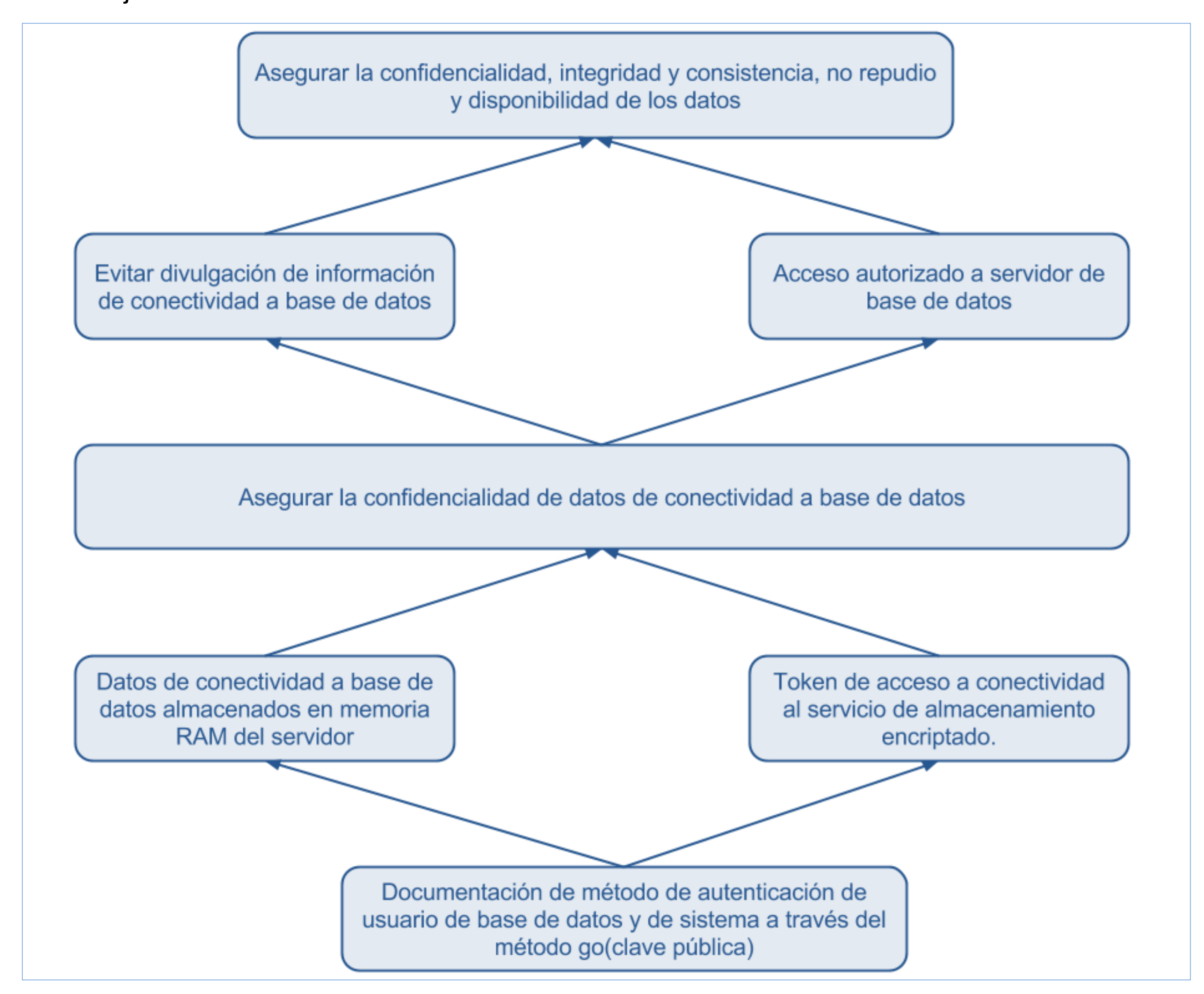


Figura 22: Árbol de objetivos

2.1.4.1 Objetivo General:

- Desarrollar un método de autenticación seguro de usuarios, el cual permita asegurar la confidencialidad de los datos de conectividad al servicio de almacenamiento.

2.1.4.2 Objetivos Específicos:

- Crear y diseñar un método seguro de autenticación, que permita asegurar la confidencialidad de los datos de conectividad.
- Desarrollar un sistema que implemente un método seguro de autenticación de usuarios de base de datos y de sistema.

2.1.5 Alcances y Limitaciones del Estudio.

2.1.5.1 Alcances del Estudio:

La presente investigación propone una solución basada en un método de autenticación, el cual permita asegurar la confidencialidad de los datos de conectividad al servicio del almacenamiento, así como también servir como método de autenticación de usuarios de sistema.

2.1.5.2 Limitaciones del Estudio:

La presente investigación pretende desarrollar un método que permita asegurar la confidencialidad de los datos de conectividad al servicio de almacenamiento; más no pretende desarrollar algoritmos de encriptación para usarlos en el método.

2.1.6 Potenciales Beneficiarios.

Los principales beneficiados con este estudio y la solución propuesta son: las empresas de software, los desarrolladores y todas las empresas que usan sistemas de información para el soporte de sus operaciones de negocio; pues no solo permite asegurar la confidencialidad de los datos de conectividad al servicio de almacenamiento, sino que a la vez es un método de autenticación de usuarios de base de datos y de sistema(Usuarios finales), proporcionando varias aplicaciones en la industria del software.

2.1.7 Aportes Esperados.

2.1.7.1 A la Universidad:

- ❖ Investigación de calidad que permita inspirar a otros estudiantes a generar conocimientos y desarrollar ideas propias.

2.1.7.2 A la Sociedad:

- ❖ Enorgullecer al Perú y mostrarles que los estudiantes investigan y tienen vocación científica.

2.1.7.3 Al Sector Productivo:

- ❖ Disminuir el riesgo de vulnerar sus datos por emplear métodos de conectividad inseguros.

2.1.7.4 Productos Tangibles de Investigación:

- ❖ Método seguro de autenticación de usuarios basado en token público.

2.2 Marco Teórico.

2.2.1 Antecedentes del Problema

2.2.1.1 Patente US 2013/0246796 A1

Información de Patente:

Título:

SYSTEM AND METHOD FOR SECURING DATABASE ACTIVITY.

Inventores:

Aviad Lichtenstadt, Tel Aviv (IL); Guy Lichtman, HertZliya (IL); Slavik Markovich, Los Altos, CA (US)

Cesionario:

McAfee, Inc.

Fecha de Publicación:

Sep. 19, 2013

Abstract:

A method is provided in one example embodiment that includes detecting database activity associated with a statement having a signature, validating the signature; and evaluating the statement as a signed statement if the signature is valid. In more particular embodiments, the signature may include a unique script identifier and a hash function of a shared key. In yet other embodiments, validating the signature may include checking a session variable and comparing the statement to a list of signed statements.

(Traducción: Un método es proporcionado en un ejemplo de realización que incluye detección de actividad de base de datos asociada con una sentencia que tiene una firma, validando la firma; y evaluando la sentencia como un declaración firmada si la firma es válida. En realizaciones más particulares, la firma puede incluir un script único de identificación y una función hash de clave compartida. En otras realizaciones, la validación de la firma puede incluir la comprobación de una variable de sesión y comparar la sentencia con una lista de sentencias firmadas.)

TECHNICAL FIELD

This specification relates in general to information systems, and more particularly, to a system and method for securing database activity.

(Traducción: La especificación se refiere en general a sistemas de información y más en particular, a un sistema y método para asegurar la actividad de base de datos.)

BACKGROUND

Databases and their database management system (DBMS) counterparts have evolved to facilitate designing, building, and maintaining complex information systems, but databases and

DBMSs themselves have also evolved into quite complex systems. The size, capabilities, and performance of databases and DBMSs have grown by orders of magnitude along with the progress of technology in the areas of processors, computer memory, computer storage, and computer networks.

Databases often contain sensitive data, which usually resides within tables that can be used by various applications. Many types of controls can be implemented to protect databases (and the sensitive data within them). For example, network security measures may be used to guard against unauthorized external access, while database activity monitoring tools can be used to audit internal activity and guard against authorized access for unauthorized purposes. However, these controls may not be able to differentiate adequately between authorized and unauthorized access to sensitive data by an authorized user. A manual investigation is often necessary to determine if many activities are approved. Thus, significant challenges remain for designing and implementing systems and methods for securing database activity.

(**Traducción**: Las bases de datos y sus sistemas de administración de base de datos han evolucionado para facilitar el diseño, la construcción y mantenimiento de complejos sistemas de información, pero las base de datos y sus dbms se han convertido en sistemas muy complejos. El tamaño, la capacidad y el rendimiento de las bases de datos y dbms han crecido por orden de magnitud junto con el progreso de la tecnología en áreas de procesadores, memoria de la computadora y red de computadoras.

Las bases de datos a menudo contienen datos sensibles, los cuales residen en tablas que pueden ser usadas por varias aplicaciones. Muchos tipos de controles son implementados para proteger las bases de datos (y los datos sensibles dentro de ellos). Por ejemplo, las medidas de seguridad de red se pueden

usar para protegerse contra el acceso externo no autorizado, mientras las herramientas de monitoreo de actividad de base de datos pueden ser usadas para auditar actividad interna y proteger contra accesos no autorizados para propósitos no autorizados. Sin embargo, estos controles pueden no ser capaces de diferenciar entre accesos autorizados y no autorizados a datos sensibles por un usuario autorizado. Una investigación manual es necesaria a menudo para determinar si muchas actividades son aprobadas. Por lo tanto, sigue habiendo retos importantes para el diseño y la implementación de sistemas y métodos para asegurar la actividad de base de datos.)

2.2.2 Base Teórica

2.2.2.1 Base de datos:

(Date, 2001) Un sistema de base de datos básicamente un sistema computarizado para guardar registros; es decir es un sistema computarizado cuya finalidad general es almacenar información y permitir a los usuarios recuperar y actualizar esa información con base en peticiones.

(Pons, Marín, Medina, Acid, & Vila, 2009) Fondo común de información almacenada en una computadora para que cualquier persona o programa autorizado pueda acceder a ella, independiente de su procedencia y del uso que haga.

(Sitio web de University System of Georgia) Una base de datos es una colección de información organizada para proporcionar información eficiente. La información recopilada podría estar en cualquier número de formatos (Electrónicos, impresos, gráficos, audio, etc).

La definición de Base de datos en cada una de las bibliografías encontradas es distinta, pero es claro que todas concuerdan en que una base de datos es un almacén de datos organizados para

poder ser recuperados por personas o entes autorizados, en esta investigación una base de datos es un sistema computarizado que almacena datos de forma organizada para ser accedidos por usuarios y aplicaciones autorizadas (Figura 23).

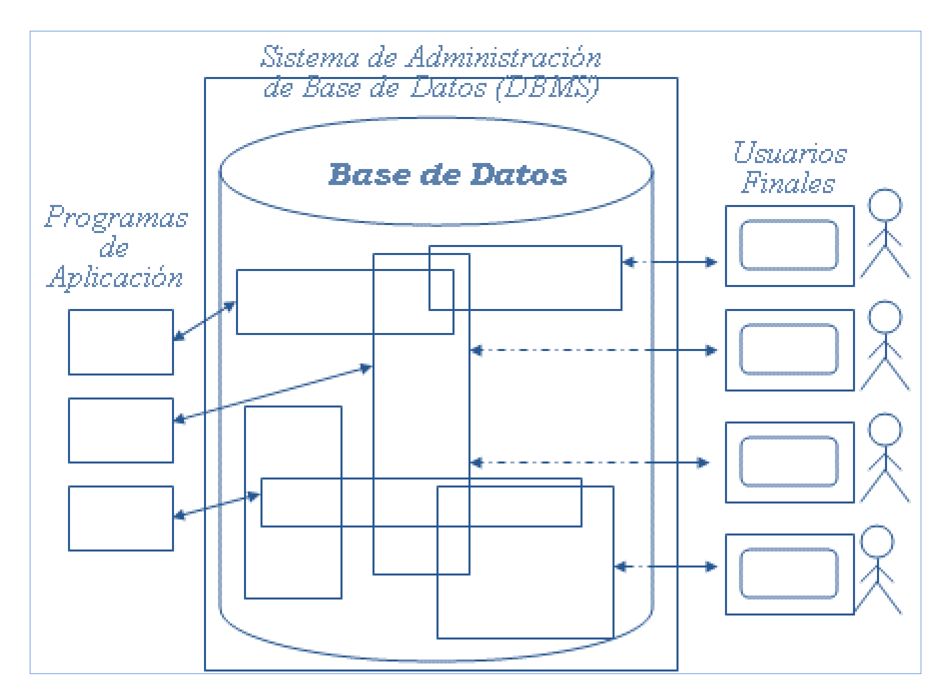


Figura 23: Base de datos

2.2.2.2 DBMS

(Date, 2001) Entre la base de datos física, es decir, los datos como están almacenados físicamente y los usuarios del sistema se encuentra una capa de software conocida por sus siglas en inglés como BDMS que traducido al español es Sistema de administración de Base de Datos, este componente se hace cargo de todas la solicitudes a la base de datos, el objetivo general del DBMS es ocultar a los usuarios de base de datos los detalles a nivel de hardware (Figura 24).

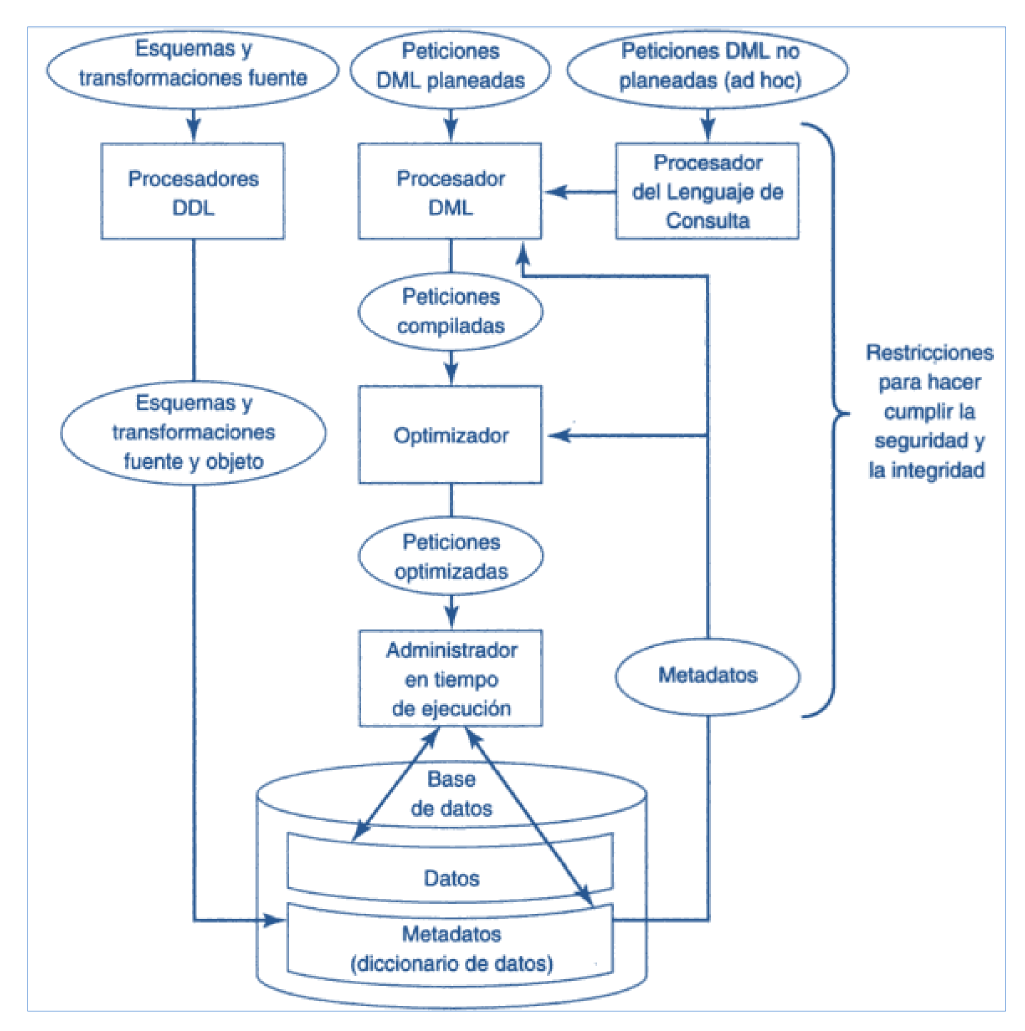


Figura 24: DBMS

2.2.2.3 Trazas SQL.

Las trazas sql es uno de los mecanismos para monitorear y auditar bases de datos, son varios los motivos que llevan a analizar una instancia de base de datos, entre los más comunes están los siguientes:

- Existen consultas que se ejecutan muy lento o que emplean una gran cantidad de recursos (memoria, disco, etc).
- Es necesario supervisar las actividades de los usuarios y las aplicaciones para comprobar el uso que hacen de la instancia.

Las trazas se configuran indicando entre otras cosas los sucesos que interesan ser analizados. Existe una gran variedad de ellos, pero no siempre interesa recoger información de todos, ya que el exceso de información puede complicar el análisis. Tenga en cuenta que la generación de trazas supone una carga adicional para el servidor y puede perjudicar el rendimiento de la instancia.

2.2.2.4 Sesiones.

(Black, 1983) Se llama sesión a la comunicación entre dos componentes de la red. La sesión puede ser de diversos tipos. Por ejemplo, puede existir una sesión entre dos operadores de dos terminales de red; puede existir una sesión entre ordenadores; asimismo puede existir una sesión entre dos programas o aplicaciones ofimáticas.

2.2.2.5 Usuarios de Sistema.

Son aquellos usuarios que puede autenticar ante un sistema de cualquier índole mediante el uso de una clave secreta, que permita identificar su existencia y autorización para el uso del sistema.

2.2.2.6 Usuario de base de datos.

Son aquellos usuarios que permiten manipular los datos (INSERT, UPDATE, DELETE, SELECT) de una base de datos y cuya autenticación debe realizarse sobre la base de datos en sí misma, quiere decir que cuando existen una autenticación de usuarios de base de datos, se abre una sesión entre la aplicación usada para autenticar y la base de datos sobre la cual se autentica.

2.2.2.7 Transacciones.

(Date, 2001) Es el proceso que obliga a llevar diferentes tareas en un solo bloque, por la necesidad de ejecución correcta de todas

las tareas, si alguna tarea falla el proceso de transacción deja todo igual a como estaba antes.

2.2.2.8 Seguridad de Información.

(ISO/IEC, 2005) Preservación de la confidencialidad, integridad y disponibilidad de la información; además también pueden estar involucradas otras propiedades como la autenticidad, responsabilidad, no repudio y confiabilidad.

2.2.2.9 Confidencialidad.

(ISO/IEC, 2005) Propiedad que rige que la información esté disponible y no sea divulgada a personas, entidad o proceso no autorizados.

2.2.2.10 Disponibilidad.

(ISO/IEC, 2005) La propiedad de estar disponible y utilizable cuando lo requiera una entidad autorizada.

2.2.2.11 Integridad.

(ISO/IEC, 2005) La propiedad de salvaguardar la exactitud.

2.2.2.12 Amenaza

(ISO/IEC, 2005) Una causa potencial de un incidente no-deseado, el cual puede resultar en daño a un sistema u organización.

2.2.2.13 Vulnerabilidad.

(ISO/IEC, 2005) La debilidad de un activo o grupo de activos que puede ser explotada por una o más amenazas.

2.2.2.14 Memoria RAM.

(Santamaria, 1993) Una memoria RAM (Ramdon Access Memory) es una memoria en la que se puede leer y escribir y en la cual los tiempos de escritura y lectura son esencialmente independientes de la localización en la que entra el dato o en la que estaba. El

dato entra en una localización determinada y se saca de esa misma localización.

2.2.3 Hipótesis

Implementar un método de autenticación que permita salvaguardar los datos de conectividad al servicio de almacenamiento en la memoria RAM del servidor, disminuye así el riesgo de robo de datos de conectividad y por ende acceso no autorizado al almacén de datos.

2.3 Marco Metodológico

La mayoría de desarrolladores no salvaguarda los datos de conectividad a los servicios de almacenamiento, su principal preocupación es conectar su aplicación con algún almacén de datos, sin importar el método usado para dicha labor.

Es deber de todo desarrollador mitigar el riesgo de obtener estos datos sin autorización, esta es la principal razón de inventar un método que permita salvaguardar los datos de conectividad y además servir como método de autenticación de usuarios.

En los siguientes aparatados se describirá la invención así como también se desarrollara una solución de software que implemente dicho método para asegurar los datos de conectividad y autenticar a los usuarios del sistema.

2.3.1 Propuesta de método de autenticación.

MÉTODO DE AUTENTICACIÓN GO

O

MÉTODO DE AUTENTICACIÓN DE CLAVE PÚBLICA.

RESUMEN

Método de autenticación de clave pública, es un método que salvaguarda los datos de conectividad a los almacenes de datos, permitiendo a los desarrolladores conectar sus aplicaciones a los servicios de almacenamiento, sin tener que preocuparse de exponer los datos en archivos físicos o embebiendo los datos de conectividad en el código fuente. Además de proporcionar un modelo de autenticación de usuarios en 3 niveles, ideal para grupos corporativos que manejan varias unidades de negocio, pues centraliza la información de los usuarios y permite monitorear la actividad SQL de los usuarios del sistema desarrollado.

OBJETIVOS DEL METODO

- Proteger los datos de conectividad a los almacenes de datos.
- Brindar un modelo de autenticación de usuarios en 3 niveles.
- Permitir monitorear la actividad SQL de cada uno de los usuarios del sistema que implementa el “METODO DE AUTENTICACIÓN DE CLAVE PUBLICA”.
- Permitir una administración centralizada de usuarios del sistema.
- Permitir realizar balanceo de carga.
- Asegurar NO REPUDIO de parte de los usuarios del sistema.

COMPONENTES

1. **Usuario SuperGoAdmin:**

   El usuario “SuperGoAdmin” auténtica sus credenciales a nivel base de datos, proporcionando los datos de conectividad al almacén de datos. Este usuario permite que los usuarios “GoAdmin” autentiquen ante el sistema mediante el uso de la clave pública que genera el usuario “SuperGoAdmin” al momento de la autenticación, por esta razón es obligatorio que antes de autenticar un usuario “GoAdmin” el usuario “SuperGoAdmin” ya este previamente autenticado.

   El usuario “SuperGoAdmin” cuenta con un almacén de datos que lleva su mismo nombre, en el cual existen tablas que permiten guardar las credenciales de los usuarios “GoAdmin” y “GoUser”, proporcionado de esta manera una administración centralizada de usuarios.

2. **Usuario GoAdmin:**

   El usuario “GoAdmin” auténtica sus credenciales a nivel de sistema y de base de datos, siendo sus credenciales de sistema comparadas en la tabla de usuarios del almacén de datos del usuario “SuperGoAdmin”, con el objetivo de

intercambiar estos datos por datos de conectividad al almacén de datos de “GoAdmin”, la cual lleva su mismo nombre.

Este usuario permite que los usuarios “GoUser” autentiquen ante el sistema mediante el uso de la clave pública que genera el usuario “GoAdmin” al momento de la autenticación, por esta razón es obligatorio que antes de autenticar un usuario “GoUser” el usuario “GoAdmin” ya este previamente autenticado.

El usuario “GoAdmin” cuenta con un almacén de datos que lleva su mismo nombre, en el cual existen tablas que permiten guardar las credenciales de los usuarios “GoUser”, lo cuales tienen acceso a realizar operaciones DML (lenguaje de manipulación de datos) sobre las tablas que soportan las reglas de negocio de una empresa, se podría decir que una base de datos del usuario “GoAdmin” es una base de datos de negocio, sobre la cual se desarrolla un sistema de información.

3. **Usuario GoUser:**

   El usuario “GoUser” auténtica sus credenciales a nivel de sistema y de base de datos, siendo sus credenciales de sistema comparadas en la tabla de usuarios del almacén de datos del usuario “GoAdmin”, con el objetivo de intercambiar estos datos por datos de conectividad al almacén de datos de “GoAdmin”.

   Los usuarios “GoUser” son considerados los usuarios finales de un sistema de información, no cuenta con un almacén de datos, ya que estos tienen privilegios de lectura y escritura sobre objetos de la base de datos de los usuarios tipo “GoAdmin”.

4. **Datos de conectividad:**

   Son aquellos datos que usa un desarrollador para conectar su aplicación a un determinado almacén de datos (Ejm: Mysql, Postgres, Oracle, etc.)

5. **Clave privada:**

   Es aquella clave que debe mantener en secreto el usuario al que se le asigno, para evitar acceso no autorizado a su cuenta o iniciar sesión ante el sistema (Refiere a sistema de base de datos o sistema de información).

6. **Clave Publica:**

   Es aquella clave pública que se le proporciona a otro usuario autorizado, con el único fin de permitirle usar su sesión o conexión con la base de datos para así intentar autenticar ante el sistema (Refiere a sistema de base de datos o sistema de información).

7. **Clave No Repudio:**

   Permite identificar tu conexión con la base de datos y así hacer una inspección manual cuando exista alguna anomalía de doble acceso al sistema desde diferentes puntos de red.

   El procedimiento manual sería preguntar al usuario autorizado que clave no repudio ha utilizado al momento de su autenticación, identificando así cual es la conexión anómala al sistema, permitiendo de esta manera identificar al intruso y así tomar las medidas necesarias.

8. **Dispositivos Tecnológicos:**

   Son aquellos dispositivos o terminales que permiten conectar a la red WAN o LAN, los dispositivos más usados para manipular sistemas informáticos son las computadoras personales, tabletas y celulares.

9. **Interfaz de entrada de datos:**

   Es aquella interfaz que permite la entrada y salida de datos.

   Ejemplo: Interfaz gráfica de autenticación de usuarios.

10. **Algoritmo de encriptación:**

    Son algoritmos que permiten cifrar la Información, para mantenerla segura de usuarios no autorizados.

11. **Memoria RAM:**

    La memoria RAM permite almacenar datos o programas que se ejecutan en los sistemas operativos.

12. **Listas Clave – Valor:**

    Son matrices tamaño nx2, n filas y 2 columnas. En la primera columna guarda la

clave y en la segunda columna el valor, de manera que el valor solo puede ser accedido mediante la clave, en los lenguajes de programación existen estos tipos de lista.

Ejemplo:

CLAVE	VALOR
45329234	JONATHAN TORRES/M/27

13. Almacén de datos:

Estructura que permite persistir los datos en el tiempo

DESCRIPCION

El método de autenticación GO o de clave pública se ha representado en tres imágenes; La figura 26 muestra la autenticación de los usuarios “SuperGoAdmin” (U1), la figura 27 muestra la autenticación de los usuarios “GoAdmin” (U2) y la figura 28 muestra la autenticación de los usuarios “GoUser” (U3).

El método de autenticación GO permite a los desarrolladores crear sus sistemas de información sin exponer los datos de conectividad (D1) a los almacenes de datos (B1, B2, B3), así como también permitir la autenticación de usuarios finales (U3). En este método se envían los datos de conectividad al servidor a través de una interfaz, para luego ser verificada y abrir sesión o conexión con el servidor de base de datos, todas las conexiones son almacenadas en memoria RAM del servidor (R1), en un lista clave – valor.

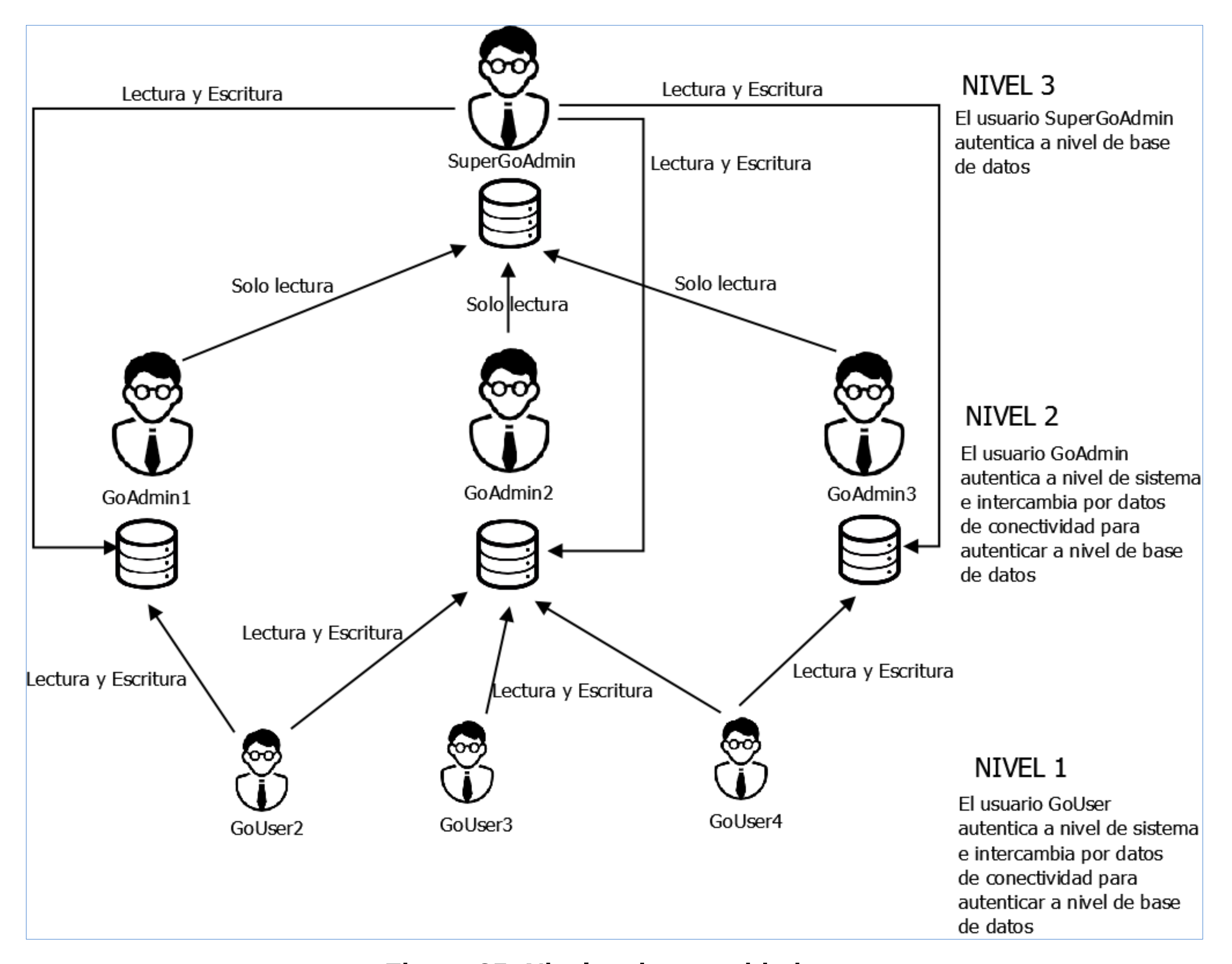


Figura 25: Niveles de seguridad

En la figura 25 se observa los 3 niveles de seguridad del método de autenticación GO, el método empieza desde el nivel más alto y el de mayor riesgo, por esta razón se recomienda hacer la autenticación en un entorno seguro, si fuera posible desde el propio servidor de aplicaciones (Servidor que contiene nuestra aplicación ejecutándose); en la figura también notamos que los usuarios GoUser (U3) no son dueños de ningún almacén de datos, así que en los GoUser (U3) no es necesario tener un usuario de base de datos por cada uno, disminuyendo de esta manera el número de conexione en el servidor de base de datos, pero si se necesita monitorear la actividad SQL del usuario GoUser, es necesario que este asignado a un usuario de base de datos.

A continuación se explica el método de autenticación GO:

Autenticación del usuario SuperGoAdmin:

Figura 26: Autenticación SuperGoAdmin

En la Figura 25 se ilustra con un flujo grama como el usuario “SuperGoAdmin” (U1) auténtica ante el sistema y abre una sesión o conexión al almacén de datos.

El usuario “SuperGoAdmin” (U1) envía los datos de conectividad(D1) (dentro de estos datos se encuentra la clave privada (KS1)) al almacén de datos (B1) junto con una clave pública (KP1) a través de dispositivos tecnológicos (DT1) que se encuentran conectados a la red WAN O LAN que se comunica con el servidor de aplicaciones que contiene el Sistema (Sistema informático que implementa método de autenticación GO), esta comunicación se realiza a través de una interfaz(I1) que permite el envío de los datos (D1 Y KP1) y así enviarlos a través del protocolo Http hasta el servidor de aplicaciones, donde los datos de conectividad (D1) son leídos; recuperada la clave pública del usuario (KP1) “SuperGoAdmin” (U1) esta se encripta mediante algún algoritmo de encriptación para luego ser consulta en una lista clave – valor (CV1) que se encuentra almacenada en la memoria RAM (R1) del servidor de aplicaciones;

Si existe la clave en la lista, es porque el usuario “SuperGoAdmin” (U1) ya había autenticado correctamente antes, entonces se convierte a hexadecimal la clave pública encriptado con el objetivo de mantener la integridad de la clave pública en los viajes que se realiza del cliente al servidor; luego encriptamos el usuario y nombre de base de datos mediante algún algoritmo de encriptación para que después se conviertan a código hexadecimal; Posterior a esta actividad almacenamos la clave pública, el usuario y nombre de base datos anteriormente convertidos a hexadecimal en la sesión de usuario del servidor y redirigir al usuario “SuperGoAdmin” (U1) a la interfaz de administración y brindarle un saludo de bienvenida al sistema (M2), ya que su autenticación fue correcta; Si no existe la clave pública encriptada en la lista clave-valor (CV1) almacenada en memoria RAM (R1) del servidor de aplicaciones se procede a verificar la conectividad al almacén de datos haciendo uso de los datos de conectividad (D1) anteriormente leídos y la tecnología competente que se necesita para esta labor; si no existe conexión a base de datos “Super-GoAdmin” (U1) enviar mensaje de error (M1) y mostrar a través de la interfaz, pero si existe conexión al almacén de datos se procede a almacenar en lista clave – valor (CV1) el objeto de conectividad, el cual solo podrá ser recuperado con clave pública encriptada; luego se vuelve a ejecutar el subproceso de almacenar token en el servidor de aplicaciones.

Autenticación del usuario GoAdmin:

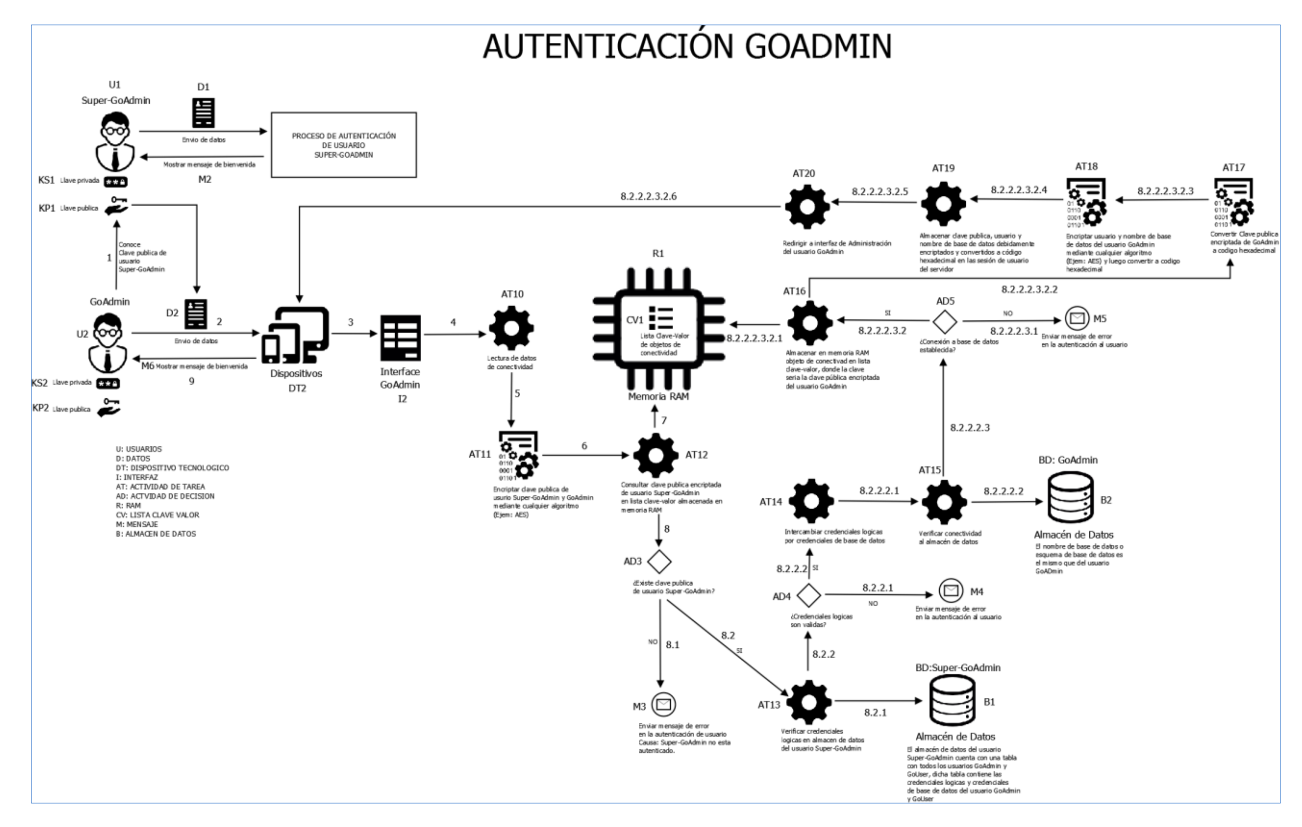


Figura 27: Autenticación GoAdmin

En la Figura 26 se ilustra con un flujo grama como el usuario “GoAdmin” (U2) auténtica ante el sistema y abre una sesión o conexión al almacén de datos.

Una vez autenticado el usuario “SuperGoAdmin” (U1), este comunica su clave pública (KP1) a los usuarios “GoAdmin” (U2); con el propósito que el usuario “GoAdmin” (U2) envié los datos de autenticación (datos lógicos comparados en la tabla de usuarios del almacén de datos del usuario “SuperGoAdmin” (U1) que serán intercambiados por datos de conectividad pertenecientes al usuario “GoAdmin” (U2)) junto con su clave pública (KP2) y la clave pública (KP1) del usuario “SuperGoAdmin” a través de dispositivos tecnológicos (DT2) que se encuentran conectados a la red WAN O LAN que se comunica con el servidor de aplicaciones que contiene el Sistema (Sistema informático que implementa método de autenticación GO), esta comunicación se realiza a través de una interfaz(I2) que permite el envío de los datos (D2, KP1 y KP2) y así enviarlos a través del protocolo Http hasta el servidor de aplicaciones, donde los datos

lógicos (D2) son leídos; recuperada la clave pública (KP1) del usuario “SuperGoAdmin” (U1) y la clave pública (KP2) del usuario “GoAdmin”, estas se encriptan mediante algún algoritmo de encriptación para luego consultar la clave pública (KP1) del usuario “SuperGoAdmin” (U1) en la lista clave – valor (CV1) que se encuentra almacenada en la memoria RAM (R1) del servidor de aplicaciones; Si no existe la clave pública (KP1) del usuario “SuperGoAdmin” (U1) enviar mensaje (M3) de error en la autenticación del usuario “GoAdmin” (U2) a causa que el usuario “SuperGoAdmin” no está autenticado; Si al contrario existiese la clave pública (KP1) en la lista clave – valor (CV1) en la memoria RAM (R1) del servidor de aplicaciones, entonces se verificarían las credenciales lógicas (D2) en la tabla de usuarios del almacén de datos (B1) del usuario “SuperGoAdmin” (U1); Si las credenciales lógicas (D2) no son válidas se envía un mensaje (M4) informando del error en la autenticación del usuarios “GoAdmin” (U2); Si por el contrario las credenciales lógicas (D2) son correctas se extrae los datos de conectividad para el usuario “GoAdmin” para luego verificar la conectividad al almacén de datos (B2) del cual es dueño el usuario “GoAdmin” (U2) y además lleva el mismo nombre; Si el usuario “GoAdmin” (U2) no logra establecer conexión con su almacén de datos se enviará un mensaje de error (M5) informando del error al autenticar a nivel de base de datos; Por el contrario, si el usuario “GoAdmin” (U2) autenticara correctamente entonces se almacenaría en la lista clave – valor (CV1) de la memoria RAM (R1) del servidor de aplicaciones el objeto conexión del usuario “GoAdmin”, donde la clave seria la clave pública encriptada del usuario “GoAdmin” (U2); una vez almacena la clave pública encriptada en memoria RAM (R1), esta es convertida a código hexadecimal; Luego encriptamos usuario y nombre de base de datos del usuario “GoAdmin” (U2) mediante cualquier algoritmo de encriptación y luego convertimos esos valores encriptados a código hexadecimal para después almacenar en la sesión de usuario del servidor de aplicaciones la clave pública encriptada y convertida a hexadecimal del usuario “GoAdmin” (U2), así como el usuario y nombre de base de datos que fueron convertidos a código hexadecimal que previamente se habían encriptado; luego se redirige al usuario “GoAdmin” (U2) a la interfaz de administración de usuarios tipo “GoAdmin” (U2) mostrando el mensaje (M6) de bienvenida.

Autenticación del usuario GoUser:

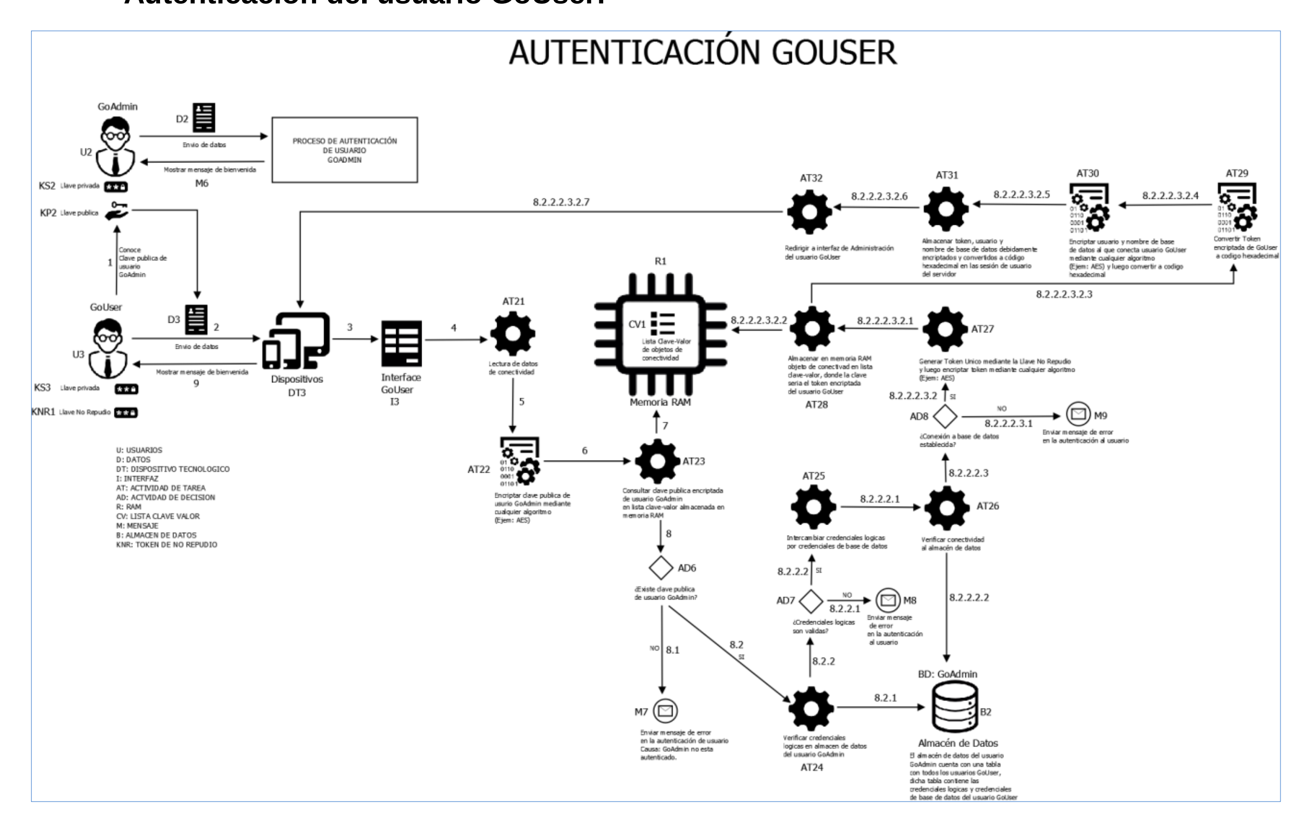


Figura 28: Autenticación GoUser

En la Figura 27 se ilustra con un flujo grama como el usuario "GoUser" (U3) auténtica ante el sistema y abre una sesión o conexión al almacén de datos.

Una vez autenticado el usuario "GoAdmin" (U2), este comunica su clave pública (KP1) a los usuarios "GoAdmin" (U3); con el propósito que el usuario "GoUser" (U3) envié los datos de autenticación (datos lógicos comparados en la tabla de usuarios del almacén de datos del usuario "GoAdmin" (U2) que serán intercambiados por datos de conectividad pertenecientes al usuario "GoUser" (U3)) junto con su clave de no repudio (KNR1) y la clave pública (KP2) del usuario "GoAdmin" a través de dispositivos tecnológicos (DT3) que se encuentran conectados a la red WAN O LAN que se comunica con el servidor de aplicaciones que contiene el Sistema (Sistema informático que implementa método de autenticación GO), esta comunicación se realiza a través de una interfaz(I3) que permite el envío de los datos (D3, KP2 y KNR1) y así enviarlos a través del protocolo Http hasta el servidor de aplicaciones, donde los datos lógicos (D3) son leídos; recuperada la clave pública (KP2) del usuario

“GoAdmin” (U2) y la clave de no repudio (KNR1) del usuario “GoUser”, estas se encriptan mediante algún algoritmo de encriptación para luego consultar la clave pública (KP2) del usuario “GoAdmin” (U2) en la lista clave – valor (CV1) que se encuentra almacenada en la memoria RAM (R1) del servidor de aplicaciones; Si no existe la clave pública (KP2) del usuario “GoAdmin” (U2) enviar mensaje (M7) de error en la autenticación del usuario “GoUser” (U3) a causa que el usuario “GoAdmin” (U2) no está autenticado; Si al contrario existiese la clave pública (KP2) en la lista clave – valor (CV1) en la memoria RAM (R1) del servidor de aplicaciones, entonces se verificarían las credenciales lógicas (D3) en la tabla de usuarios del almacén de datos (B2) del usuario “GoAdmin” (U2); Si las credenciales lógicas (D3) no son válidas se envía un mensaje (M8) informando del error en la autenticación del usuario “GoUser” (U3); Si por el contrario las credenciales lógicas (D3) son correctas se extrae los datos de conectividad para el usuario “GoUser” (U3) para luego verificar la conectividad al almacén de datos (B2) al cual tiene acceso de lectura y escritura; Si el usuario “GoUser” (U3) no logra establecer conexión con su almacén de datos se enviará un mensaje de error (M9) informando del error al autenticar a nivel de base de datos; Por el contrario, si el usuario “GoUser” (U3) autenticara correctamente entonces se almacenaría en la lista clave – valor (CV1) de la memoria RAM (R1) del servidor de aplicaciones el objeto conexión del usuario “GoUser”, donde la clave seria la clave de no repudio encriptada del usuario “GoUser”; una vez almacena la clave de no repudio que fue encriptada y almacenada memoria RAM (R1), esta es convertida a código hexadecimal; Luego encriptamos usuario y nombre de base de datos del al que el usuario “GoUser” (U3) conectó mediante cualquier algoritmo de encriptación y luego convertimos esos valores encriptados a código hexadecimal para después almacenar en la sesión de usuario del servidor de aplicaciones, la clave de no repudio encriptada y convertida a hexadecimal del usuario “GoUser” (U3), así como el usuario y nombre de base de datos que fueron convertidos a código hexadecimal que previamente se habían encriptado; luego se redirige al usuario “GoUser” (U2) a la interfaz de sistema de usuarios tipo “GoUser” (U3) mostrando el mensaje (M9) de bienvenida.

RECLAMACIONES

- Salvaguardar los datos de conectividad a los servicios de almacenamiento.

    - Los datos de conectividad son encriptados mediante cualquier algoritmo de encriptación después de verificar si existe conectividad con el servicio de almacenamiento, para luego ser guardados en la memoria RAM del servidor de aplicaciones, y así de esta forma no exponer los datos de conectividad en archivos de texto plano o embebido en el código fuente.

- Autenticación de usuarios de sistema a nivel lógico y de base de datos.
    - El método permite autenticar a los usuarios del sistema mediante un esquema de seguridad de autenticación basado en 3 niveles.
        - Nivel 3: Es el nivel de mayor riesgo, donde son enviados los datos de conectividad mediante el protocolo Http y así establecer conexión con el almacén de datos del usuarios “SuperGoAdmin”.

        - Nivel 2: Los datos enviados aquí son lógicos, a excepción del nombre único que permite identificar al almacén de datos, con el propósito de verificar su existencia en la tabla de usuarios que está contenida en el almacén de datos del usuario “SuperGoAdmin”, para luego ser intercambiados por datos de conectividad y así establecer una conexión con el almacén de datos del usuario “GoAdmin”.

        - Nivel 1: Los datos enviados aquí son lógicos, y permiten verificar su existencia en la tabla de usuarios que está contenida en el almacén de datos del usuario “GoAdmin”, para luego ser intercambiados por datos de conectividad y así establecer una conexión con el almacén de datos del usuario “GoUser”.

- Monitoreo de actividad SQL de cada uno de los usuarios del Sistema.
    - Cada usuario está relacionado con un usuario de base de datos, lo que permite abrir una conexión por cada usuario de sistemas y así monitorear su actividad SQL de manera independiente.

- Balancear carga de las aplicaciones.

    - Al enviar los datos de conectividad y no tener almacenado dentro de la aplicación en el código fuente o en archivos de texto plano, se puede configurar la aplicación en diferentes servidores.

- Controlar el número de conexiones al servicio de almacenamiento.
    - Controlar el número de conexiones o sesiones abiertas en la base de datos nos permite controlar y dar seguimiento a la performance del servidor de base de datos y hardware de servidor.
- Usar las claves públicas de los usuarios “GoAdmin” como licencias de uso, ya que sin la clave pública de “GoAdmin” los usuarios finales o “GoUser” no podrán autenticar ante el sistema, lo que permite bajar o subir el servicio mediante el uso de la clave pública, lo mejor de todo es que la clave pública puede cambiar en el tiempo.
- Usar las clave públicas de los usuarios “SuperGoAdmin” y “GoAdmin”, así como también la clave de no repudio de los usuarios “GoUser” como token de acceso al servicio de almacenamiento en cada una de las peticiones en un arquitectura tipo SOA, es decir el uso de WebService, ya sea que esta aplique el formato de transmisión de datos REST o se base en el protocolo SOAP.

2.3.2 Aplicación de método en una aplicación web.

En el presente apartado no se pretende documentar y explicar el ciclo de vida de un desarrollo de software, sino ejemplificar la aplicación del método de autenticación de clave pública (Método de autenticación GO) en un proyecto real, y así llevar la teoría del método GO a la práctica.

Este proyecto se ha desarrollado para el Grupo Olano, el cual cuenta con muchas unidades de negocios y necesita el control y monitoreo de las operaciones comerciales en cada una de sus unidades de negocio, por esta razón el método de autenticación GO se integra completamente, ya que no solo permite brindar seguridad al autenticar, sino escalabilidad y balanceo de carga.

Empezar por explicar el patrón de diseño propuesta para las soluciones de software a desarrollar.

En la figura 29 se puede observar el arquitectura de software usada en el Grupo Olano para el desarrollo de las aplicaciones corporativas, la cual cuenta con el

RDBMS SQL Server 2008 R2 que almacena sus datos; para establecer la conexión entre el RDBMS y las aplicaciones se ha usado JDO como tecnología de persistencia a DATOS; en la programación se usó un patrón de diseño modificado a partir de MVC y MVP, para que la programación este orientada al desarrollo de procesos y servicios; se usa el Google Web Toolkit para el desarrollo del frontend en combinación con los JSP para brindar una mayor seguridad.

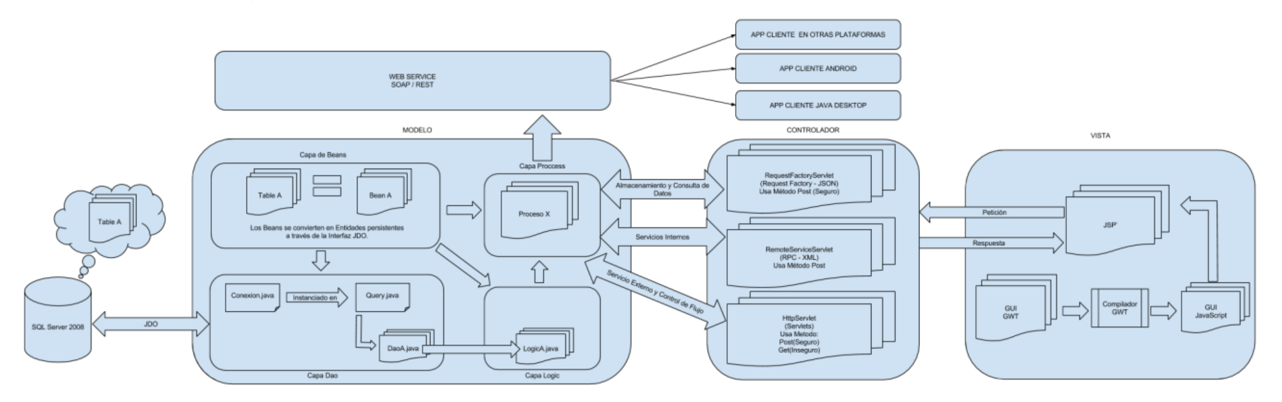


Figura 29: Arquitectura de Software

A continuación se explicará cómo se desarrolló y se aplicó el método de autenticación de clave pública (Método de autenticación GO) en un escenario real, el cual permite validar el método y verificar nuestra hipótesis, la cual afirma que guardar los datos de conectividad en la memoria RAM del servidor, disminuye el riesgo de robar la información de conectividad.

En la figura 30 se observar los niveles de seguridad que brinda el método de autenticación GO al desarrollo de aplicaciones en el Grupo Olano.

En el nivel 3 se encuentra al usuario corporativo (“SUPERGOADMIN”), el cual es dueño de una base de datos que lleva su mismo nombre, en la que se guardan de manera centralizada los usuarios “GoAdmin” y “GoUser”, permite al Grupo Olano llevar tareas administrativas y SQL sobre los usuarios del sistema, entre las tareas que se puede desarrollar con este usuarios son:

Crear nuevas empresas, lo implica que se debe crear una base de datos y usuario “GoAdmin” con los privilegios adecuados; crear usuarios de sistema “GoUser”, los cuales tendrá privilegios de escritura y lectura a las bases de datos de negocio; monitorear y dar seguimiento a la actividad SQL de cada uno de los usuarios “GoUser”, “GoAdmin” e incluso al usuario “SuperGoAdmin”.

En el nivel 2 se encuentra a los usuarios de negocio (“GoAdmin”), los cuales son dueños de las base de datos de negocio en las que se almacenan las operaciones comerciales, permitiendo descentralizar el control y monitoreo de los usuarios “GoUser”, limitando solamente al control de aquellos usuarios “GoUser” que tienen acceso a su base de datos de negocio.

El nivel 1 se encuentra a los usuarios del sistema o usuarios finales (“GoUser”), estos usuarios no son dueños de ningún almacén de datos, pero si tienen acceso a las base de datos de negocio de los usuarios “GoAdmin”.

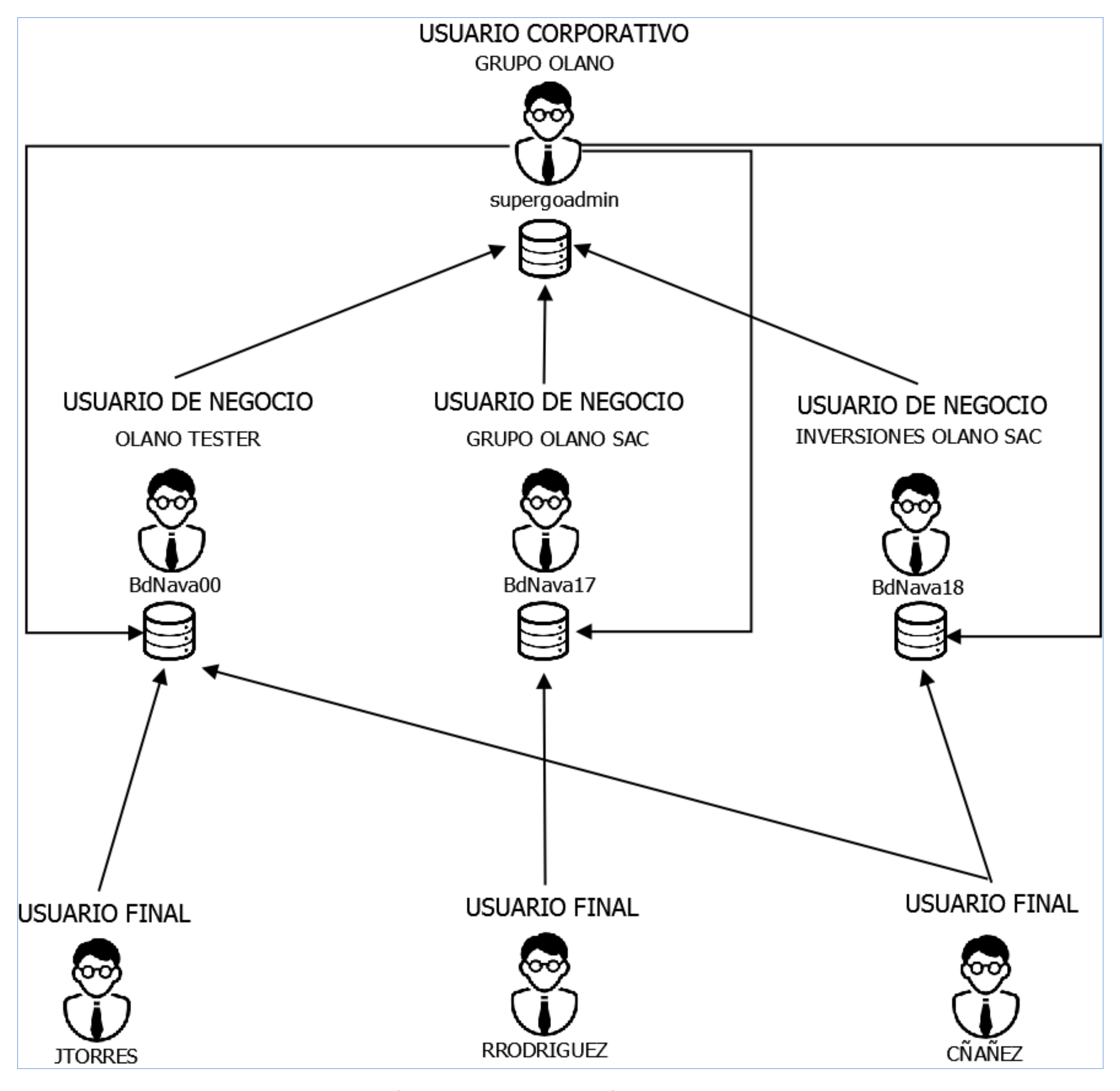


Figura 30: Método autenticación GO en Grupo Olano

Para implementar el método de autenticación de clave pública se ha tenido que proponer el siguiente diseño de base de datos.

En la figurar 31 se puede observar el diseño de base de datos que maneja el usuario “SuperGoAdmin”, el cual sirve para el registro, control y monitoreo de usuarios “GoAdmin” y “GoUser”.

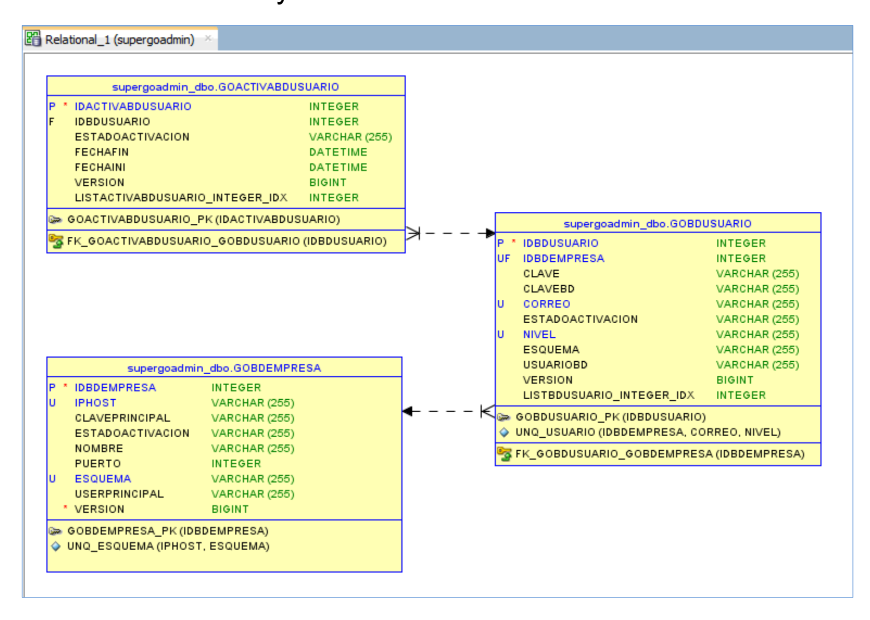


Figura 31: BD SuperGoAdmin

En la figura 32 se puede observar el diseño de base de datos para los usuarios “GoAdmin”; este diseño es una réplica del diseño de datos de usuario “SuperGoAdmin”, ya que las bases de datos de los “GoAdmin”, solo almacenan a los usuarios “GoUser.

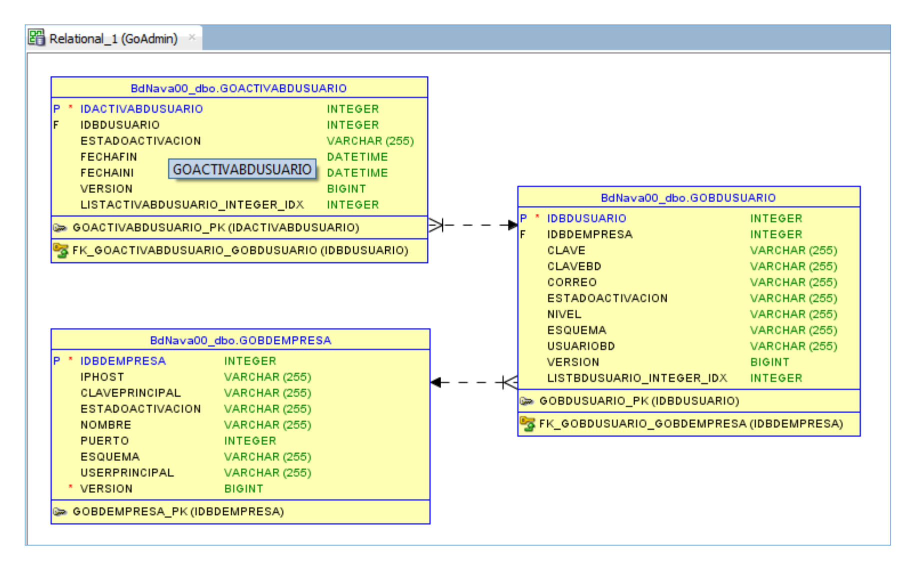


Figura 32: BD por usuario GoAdmin

El siguiente procedimiento almacenado permite crear empresas, lo cual implica crear bases de datos y usuarios "GoAdmin", si bien no se ha incluido la sentencias SQL que permitan la creación de base de datos, el procedimiento para realizarlo no es imposible, para este caso se crearon los usuarios y las base de datos "bdnava18" y "bdnava00" de manera manual y con lo adecuados privilegios de acceso (Ver manual de usuarios en anexo A de la investigación).

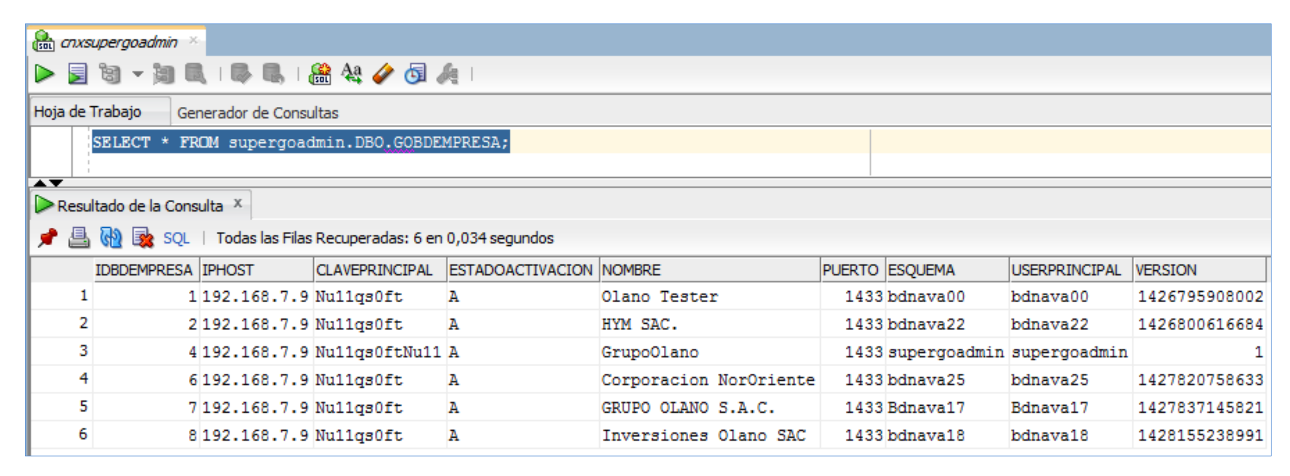
Este procedimiento permite guardar los datos de una nueva empresa tanto en la base de datos del usuario "supergoadmin" como en las base de datos de negocio de "bdnava00" y "bdnava18", se puede observar esto en las figuraras 34, 35 y 36 respectivamente.

cnxsupergoadmin

Hoja de Trabajo | Generador de Consultas

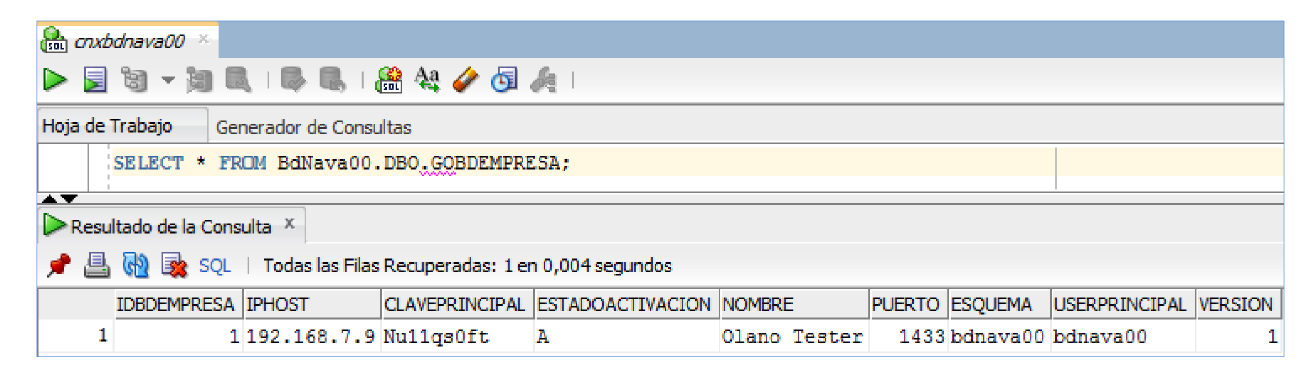
```sql
-- =============================================
-- Author:  Jonathan Franchesco Torres Baca
-- =============================================
CREATE PROCEDURE [dbo].[insertBdEmpresa]
-- Add the parameters for the stored procedure here
@idbdempresa   INT,
@nombreEmpresa VARCHAR(255),
@host          VARCHAR(255),
@puerto        INT,
@esquema       VARCHAR(255),
@usuario       VARCHAR(255),
@clave         VARCHAR(255)
AS
  DECLARE @str nvarchar(MAX)
  BEGIN
    -- SET NOCOUNT ON added to prevent extra result sets from
    -- interfering with SELECT statements.
    SET NOCOUNT ON;
    SET @str='INSERT INTO '+@esquema+
    '.DBO.GOBDEMPRESA(IDBDEMPRESA,NOMBRE,IPHOST,
    PUERTO,ESQUEMA,USERPRINCIPAL,CLAVEPRINCIPAL,VERSION,ESTADOACTIVACION) '
    SET @str=@str +'VALUES('+CAST(@idbdempresa AS VARCHAR)+','+CHAR(39)+
    @nombreEmpresa+CHAR(39)+','+CHAR(39)+@host+CHAR(39)+','+CAST(@puerto AS
    VARCHAR)+','+CHAR(39)+@esquema+CHAR(39)+','+CHAR(39)+@usuario+CHAR(39)+
    ','+CHAR(39)+@clave+CHAR(39)+','+CAST(1 AS VARCHAR)+','+CHAR(39)+'A'+
    CHAR(39)+')'
    EXEC sp_executesql @str
    SELECT
      'true' AS result
  END
```

Figura 33: procedimiento insertar empresa



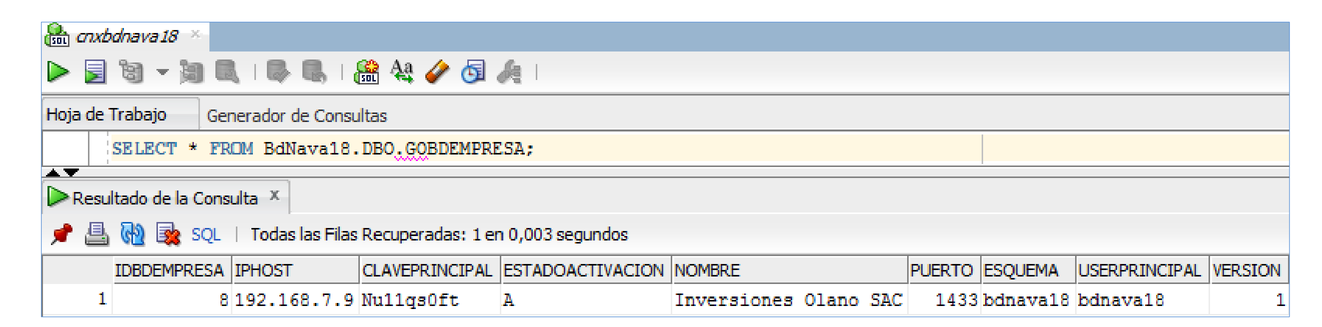
	IDBDEMPRESA	IPHOST	CLAVEPRINCIPAL	ESTADOACTIVACION	NOMBRE	PUERTO	ESQUEMA	USERPRINCIPAL	VERSION
1	1	192.168.7.9	Nu11qs0ft	A	Olano Tester	1433	bdnava00	bdnava00	1426795908002
2	2	192.168.7.9	Nu11qs0ft	A	HYM SAC.	1433	bdnava22	bdnava22	1426800616684
3	4	192.168.7.9	Nu11qs0ftNu11	A	GrupoOlano	1433	supergoadmin	supergoadmin	1
4	6	192.168.7.9	Nu11qs0ft	A	Corporacion NorOriente	1433	bdnava25	bdnava25	1427820758633
5	7	192.168.7.9	Nu11qs0ft	A	GRUPO OLANO S.A.C.	1433	Bdnava17	Bdnava17	1427837145821
6	8	192.168.7.9	Nu11qs0ft	A	Inversiones Olano SAC	1433	bdnava18	bdnava18	1428155238991

Figura 34: Tabla Empresa de Usuario supergoadmin



	IDBDEMPRESA	IPHOST	CLAVEPRINCIPAL	ESTADOACTIVACION	NOMBRE	PUERTO	ESQUEMA	USERPRINCIPAL	VERSION
1	1	192.168.7.9	Nu11qs0ft	A	Olano Tester	1433	bdnava00	bdnava00	1

Figura 35: Tabla empresa de usuario bdnava00



	IDBDEMPRESA	IPHOST	CLAVEPRINCIPAL	ESTADOACTIVACION	NOMBRE	PUERTO	ESQUEMA	USERPRINCIPAL	VERSION
1	8	192.168.7.9	Nu11qs0ft	A	Inversiones Olano SAC	1433	bdnava18	bdnava18	1

Figura 36: Tabla empresa de usuario bdnava18

En la figura 37 se puede observar el procedimiento almacenado que permite crear usuarios “GoAdmin” y “GoUser”, este procedimiento guarda información de los usuarios en la tabla “GOBDUSUARIO” y “GOACTIVABDUSUARIO” de la base de datos supergoadmin, así como también en la base de datos negocio que corresponda el usuario, ya sea “GoAdmin” o “GoUser”, los cuales pueden

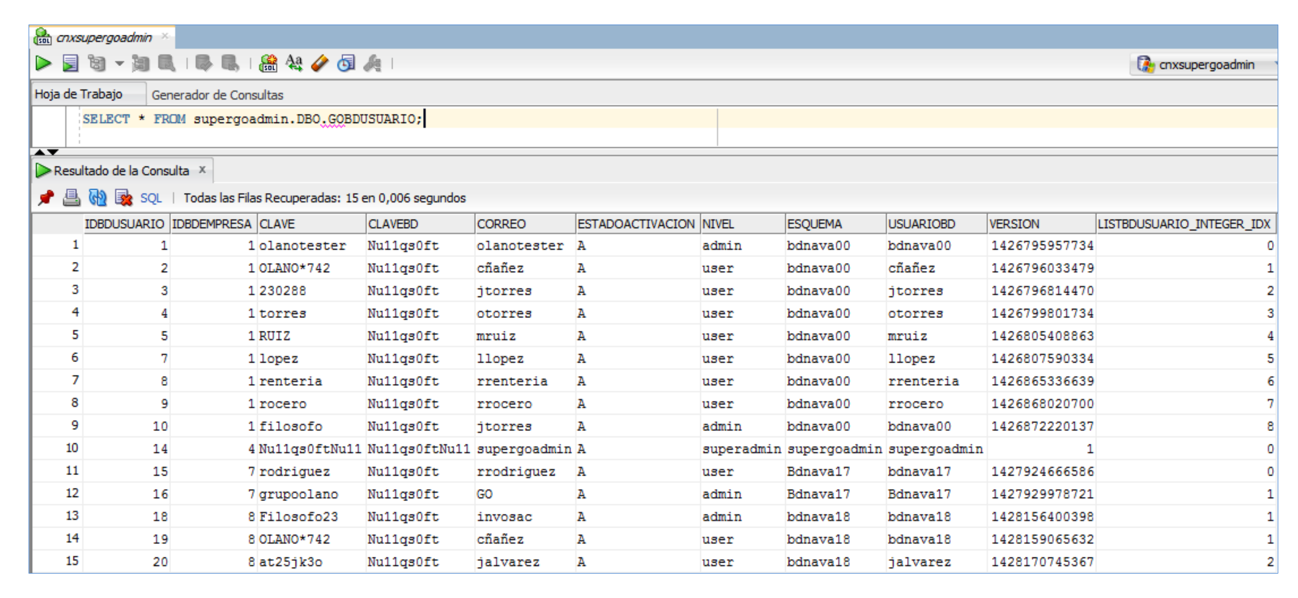
observarse en la figuras 38, 39, 40, 41, 42 y 43.

cnxsupergoadmin

Hoja de Trabajo | Generador de Consultas

```
-- ============================================
-- Author: Jonathan Franchesco Torres Baca
-- ============================================
CREATE PROCEDURE [dbo].[insertBdUsuario]
-- Add the parameters for the stored procedure here
@idbdusuario INT,
@idbdempresa INT,
@esquema     VARCHAR(255),
@nivel       VARCHAR(255),
@userlog     VARCHAR(255),
@clavelog    VARCHAR(255),
@userbd      VARCHAR(255),
@clavebd     VARCHAR(255),
@fechaini    DATETIME
AS
  DECLARE @str nvarchar(MAX)
  DECLARE @cad nvarchar(MAX)
    --declare @idBdUsuario int
  BEGIN
    -- SET NOCOUNT ON added to prevent extra result sets from
    -- interfering with SELECT statements.
    SET NOCOUNT ON;
    SET @str='INSERT INTO '+@esquema+
    '.DBO.GOBDUSUARIO(IDBDUSUARIO,IDBDEMPRESA,ESQUEMA,NIVEL,CORREO,CLAVE,
    USUARIOBD,CLAVEBD,VERSION,ESTADOACTIVACION,LISTBDUSUARIO_INTEGER_IDX) '
    SET @str=@str+'VALUES('+CAST(@idbdusuario AS VARCHAR)+','+CAST(@idbdempresa
    AS VARCHAR)+','+CHAR(39)+@esquema+CHAR(39)+','+CHAR(39)+@nivel+CHAR(39)+
    ','+CHAR(39)+@userlog+CHAR(39)+','+CHAR(39)+@clavelog+CHAR(39)+','+CHAR(39)
    +@userbd+CHAR(+','+CHAR(39)+@clavebd+CHAR(39)+','+CAST(1 AS VARCHAR)
    +','+CHAR(39)+'A'+CHAR(39)+','+CAST(0 AS VARCHAR)+')'
    EXEC sp_executesql @str
    SET @cad='use '
    SET @cad=@cad+@esquema
    EXEC sp_executesql @cad
    --set @idBdUsuario= @@IDENTITY
    SET @str='INSERT INTO '+@esquema+
    '.DBO.GOACTIVABDUSUARIO(IDBDUSUARIO,ESTADOACTIVACION,FECHAINI,
    VERSION,LISTACTIVABDUSUARIO_INTEGER_IDX) '
    SET @str=@str+'VALUES('+CAST(@idBdUsuario AS VARCHAR)+','+CHAR(39)+'A'+
    CHAR(39)+','+CHAR(39)+CAST(@fechaini AS VARCHAR)+CHAR(39)+','+
    CAST(1 AS VARCHAR)  +','+CAST(0 AS VARCHAR)+')'
    EXEC sp_executesql @str
    SELECT
      'true' AS result
  END
```

Figura 37: Procedimiento de creación de usuarios



cnxsupergoadmin

Hoja de Trabajo | Generador de Consultas

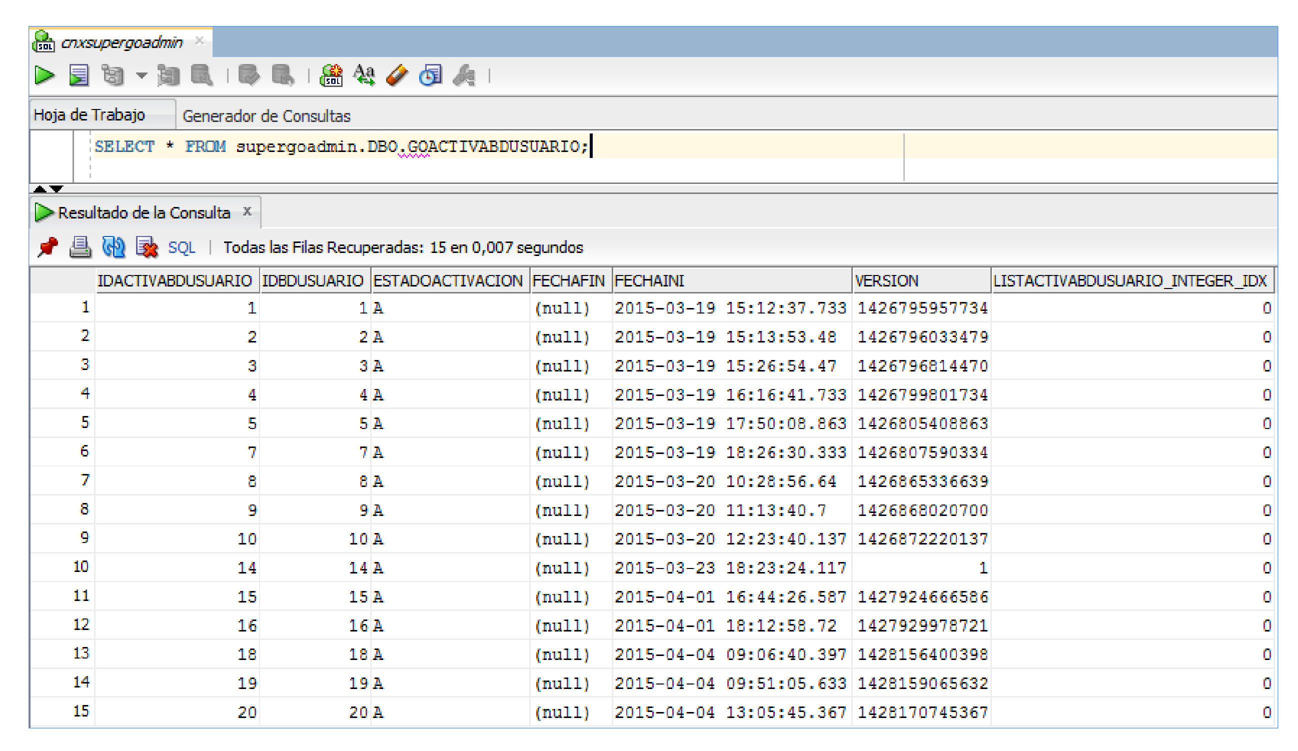
```
SELECT * FROM supergoadmin.DBO.GOBDUSUARIO;
```

Resultado de la Consulta

SQL | Todas las Filas Recuperadas: 15 en 0,006 segundos

	IDBDUSUARIO	IDBDEMPRESA	CLAVE	CLAVEBD	CORREO	ESTADOACTIVACION	NIVEL	ESQUEMA	USUARIOBD	VERSION	LISTBDUSUARIO_INTEGER_IDX
1	1	1	olanotester	Nu11qs0ft	olanotester	A	admin	bdnava00	bdnava00	1426795957734	0
2	2	1	OLANO*742	Nu11qs0ft	cñañez	A	user	bdnava00	cñañez	1426796033479	1
3	3	1	230288	Nu11qs0ft	jtorres	A	user	bdnava00	jtorres	1426796814470	2
4	4	1	torres	Nu11qs0ft	otorres	A	user	bdnava00	otorres	1426799801734	3
5	5	1	RUIZ	Nu11qs0ft	mruiz	A	user	bdnava00	mruiz	1426805408863	4
6	7	1	lopez	Nu11qs0ft	llopez	A	user	bdnava00	llopez	1426807590334	5
7	8	1	renteria	Nu11qs0ft	rrenteria	A	user	bdnava00	rrenteria	1426865336639	6
8	9	1	rocero	Nu11qs0ft	rrocero	A	user	bdnava00	rrocero	1426868020700	7
9	10	1	filosofo	Nu11qs0ft	jtorres	A	admin	bdnava00	bdnava00	1426872220137	8
10	14	4	Nu11qs0ftNu11	Nu11qs0ftNu11	supergoadmin	A	superadmin	supergoadmin	supergoadmin	1	0
11	15	7	rodriguez	Nu11qs0ft	rrodriguez	A	user	Bdnava17	bdnava17	1427924666586	0
12	16	7	grupoolano	Nu11qs0ft	GO	A	admin	Bdnava17	Bdnava17	1427929978721	1
13	18	8	Filosofo23	Nu11qs0ft	invosac	A	admin	bdnava18	bdnava18	1428156400398	1
14	19	8	OLANO*742	Nu11qs0ft	cñañez	A	user	bdnava18	bdnava18	1428159065632	1
15	20	8	at25jk3o	Nu11qs0ft	jalvarez	A	user	bdnava18	jalvarez	1428170745367	2

Figura 38: Tabla Usuarios de supergoadmin



cnxsupergoadmin

Hoja de Trabajo | Generador de Consultas

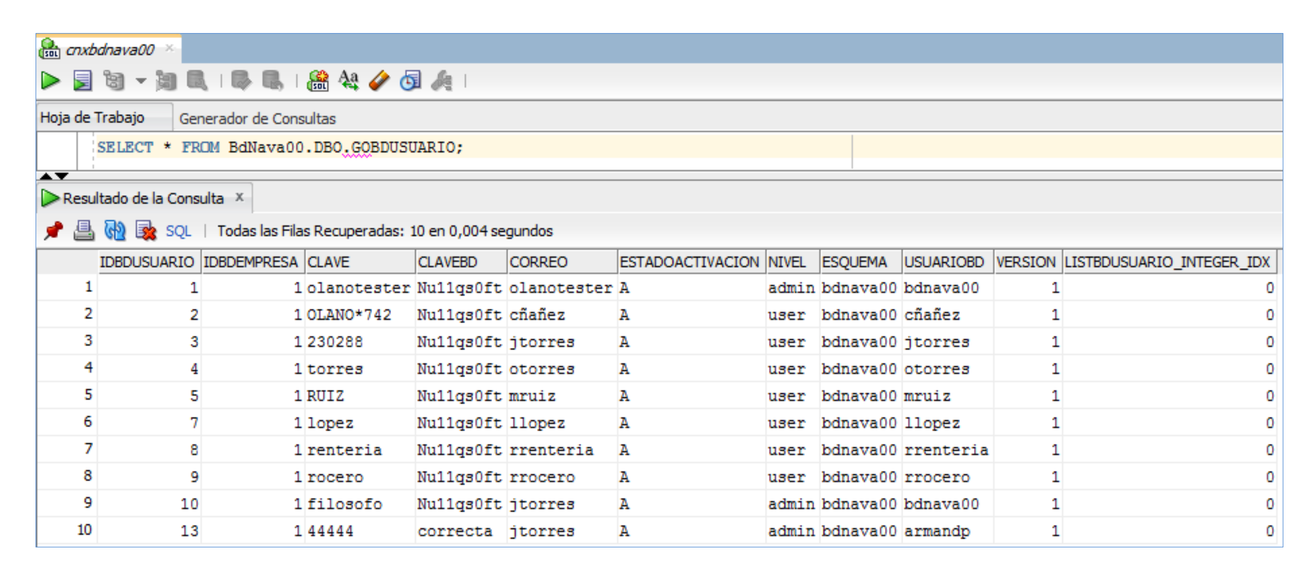
```
SELECT * FROM supergoadmin.DBO.GOACTIVABDUSUARIO;
```

Resultado de la Consulta

SQL | Todas las Filas Recuperadas: 15 en 0,007 segundos

	IDACTIVABDUSUARIO	IDBDUSUARIO	ESTADOACTIVACION	FECHAFIN	FECHAINI	VERSION	LISTACTIVABDUSUARIO_INTEGER_IDX
1	1	1	A	(null)	2015-03-19 15:12:37.733	1426795957734	0
2	2	2	A	(null)	2015-03-19 15:13:53.48	1426796033479	0
3	3	3	A	(null)	2015-03-19 15:26:54.47	1426796814470	0
4	4	4	A	(null)	2015-03-19 16:16:41.733	1426799801734	0
5	5	5	A	(null)	2015-03-19 17:50:08.863	1426805408863	0
6	7	7	A	(null)	2015-03-19 18:26:30.333	1426807590334	0
7	8	8	A	(null)	2015-03-20 10:28:56.64	1426865336639	0
8	9	9	A	(null)	2015-03-20 11:13:40.7	1426868020700	0
9	10	10	A	(null)	2015-03-20 12:23:40.137	1426872220137	0
10	14	14	A	(null)	2015-03-23 18:23:24.117	1	0
11	15	15	A	(null)	2015-04-01 16:44:26.587	1427924666586	0
12	16	16	A	(null)	2015-04-01 18:12:58.72	1427929978721	0
13	18	18	A	(null)	2015-04-04 09:06:40.397	1428156400398	0
14	19	19	A	(null)	2015-04-04 09:51:05.633	1428159065632	0
15	20	20	A	(null)	2015-04-04 13:05:45.367	1428170745367	0

Figura 39: Tabla de activación de usuarios por supergoadmin



cnxbdnava00

Hoja de Trabajo | Generador de Consultas

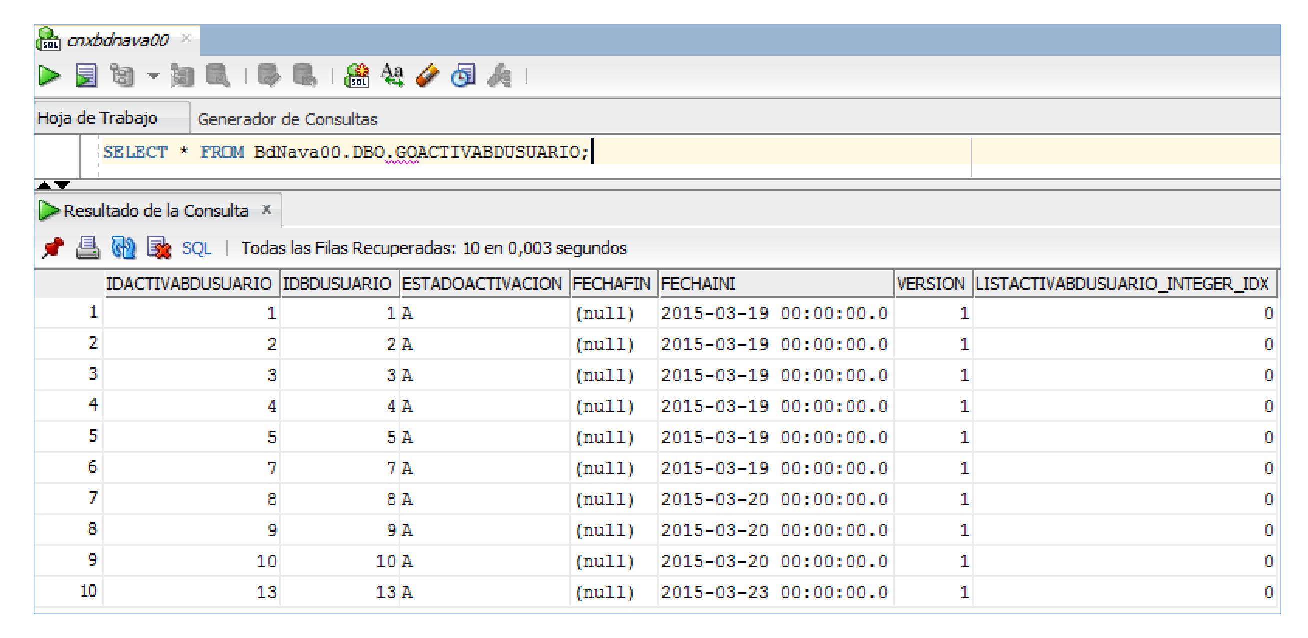
```
SELECT * FROM BdNava00.DBO.GOBDUSUARIO;
```

Resultado de la Consulta

SQL | Todas las Filas Recuperadas: 10 en 0,004 segundos

	IDBDUSUARIO	IDBDEMPRESA	CLAVE	CLAVEBD	CORREO	ESTADOACTIVACION	NIVEL	ESQUEMA	USUARIOBD	VERSION	LISTBDUSUARIO_INTEGER_IDX
1	1	1	olanotester	Nu11qs0ft	olanotester	A	admin	bdnava00	bdnava00	1	0
2	2	1	OLANO*742	Nu11qs0ft	cñañez	A	user	bdnava00	cñañez	1	0
3	3	1	230288	Nu11qs0ft	jtorres	A	user	bdnava00	jtorres	1	0
4	4	1	torres	Nu11qs0ft	otorres	A	user	bdnava00	otorres	1	0
5	5	1	RUIZ	Nu11qs0ft	mruiz	A	user	bdnava00	mruiz	1	0
6	7	1	lopez	Nu11qs0ft	llopez	A	user	bdnava00	llopez	1	0
7	8	1	renteria	Nu11qs0ft	rrenteria	A	user	bdnava00	rrenteria	1	0
8	9	1	rocero	Nu11qs0ft	rrocero	A	user	bdnava00	rrocero	1	0
9	10	1	filosofo	Nu11qs0ft	jtorres	A	admin	bdnava00	bdnava00	1	0
10	13	1	44444	correcta	jtorres	A	admin	bdnava00	armandp	1	0

Figura 40: Tabla de usuarios de bdnava00



cnxbdnava00

Hoja de Trabajo | Generador de Consultas

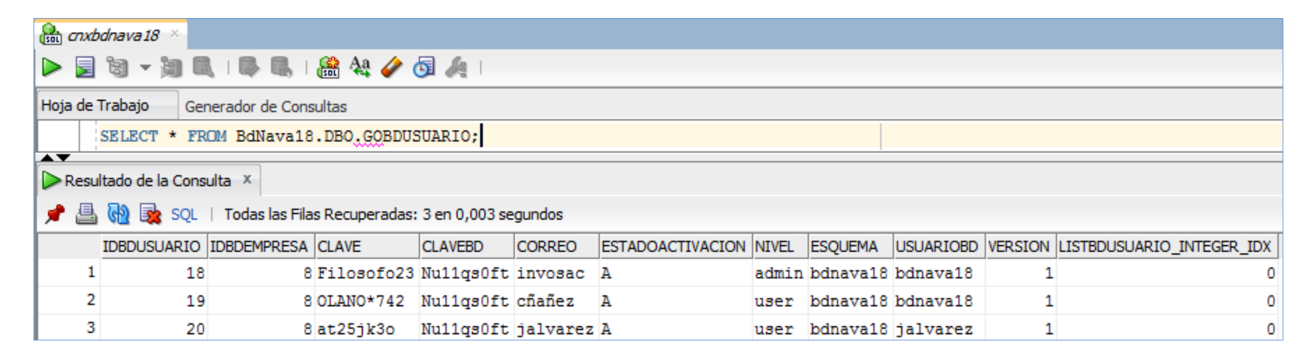
```
SELECT * FROM BdNava00.DBO.GOACTIVABDUSUARIO;
```

Resultado de la Consulta

SQL | Todas las Filas Recuperadas: 10 en 0,003 segundos

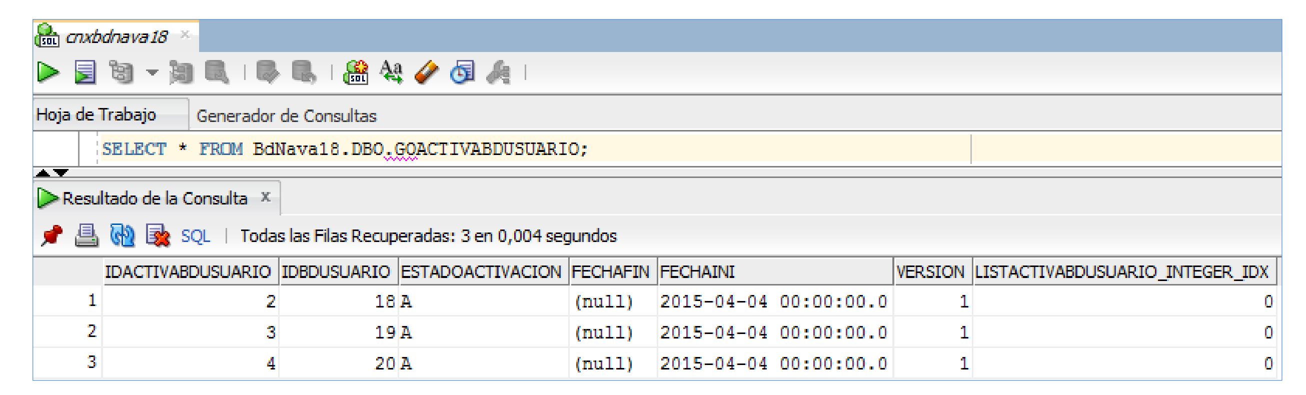
	IDACTIVABDUSUARIO	IDBDUSUARIO	ESTADOACTIVACION	FECHAFIN	FECHAINI	VERSION	LISTACTIVABDUSUARIO_INTEGER_IDX
1	1	1	A	(null)	2015-03-19 00:00:00.0	1	0
2	2	2	A	(null)	2015-03-19 00:00:00.0	1	0
3	3	3	A	(null)	2015-03-19 00:00:00.0	1	0
4	4	4	A	(null)	2015-03-19 00:00:00.0	1	0
5	5	5	A	(null)	2015-03-19 00:00:00.0	1	0
6	7	7	A	(null)	2015-03-19 00:00:00.0	1	0
7	8	8	A	(null)	2015-03-20 00:00:00.0	1	0
8	9	9	A	(null)	2015-03-20 00:00:00.0	1	0
9	10	10	A	(null)	2015-03-20 00:00:00.0	1	0
10	13	13	A	(null)	2015-03-23 00:00:00.0	1	0

Figura 41: Tabla de activación de usuarios por bdnava00



	IDBDUSUARIO	IDBDEMPRESA	CLAVE	CLAVEBD	CORREO	ESTADOACTIVACION	NIVEL	ESQUEMA	USUARIOBD	VERSION	LISTBDUSUARIO_INTEGER_IDX
1	18	8	Filosofo23	Nu11qs0ft	invosac	A	admin	bdnava18	bdnava18	1	0
2	19	8	OLANO*742	Nu11qs0ft	cñañez	A	user	bdnava18	bdnava18	1	0
3	20	8	at25jk3o	Nu11qs0ft	jalvarez	A	user	bdnava18	jalvarez	1	0

Figura 42: Tabla de usuarios de bdnava18



	IDACTIVABDUSUARIO	IDBDUSUARIO	ESTADOACTIVACION	FECHAFIN	FECHAINI	VERSION	LISTACTIVABDUSUARIO_INTEGER_IDX
1	2	18	A	(null)	2015-04-04 00:00:00.0	1	0
2	3	19	A	(null)	2015-04-04 00:00:00.0	1	0
3	4	20	A	(null)	2015-04-04 00:00:00.0	1	0

Figura 43: Tabla de activación de usuarios por bdnava18

En la figura 44 se muestra la interfaz gráfica que permite el envió de los datos de conectividad. Esta autenticación es a nivel de base de datos.

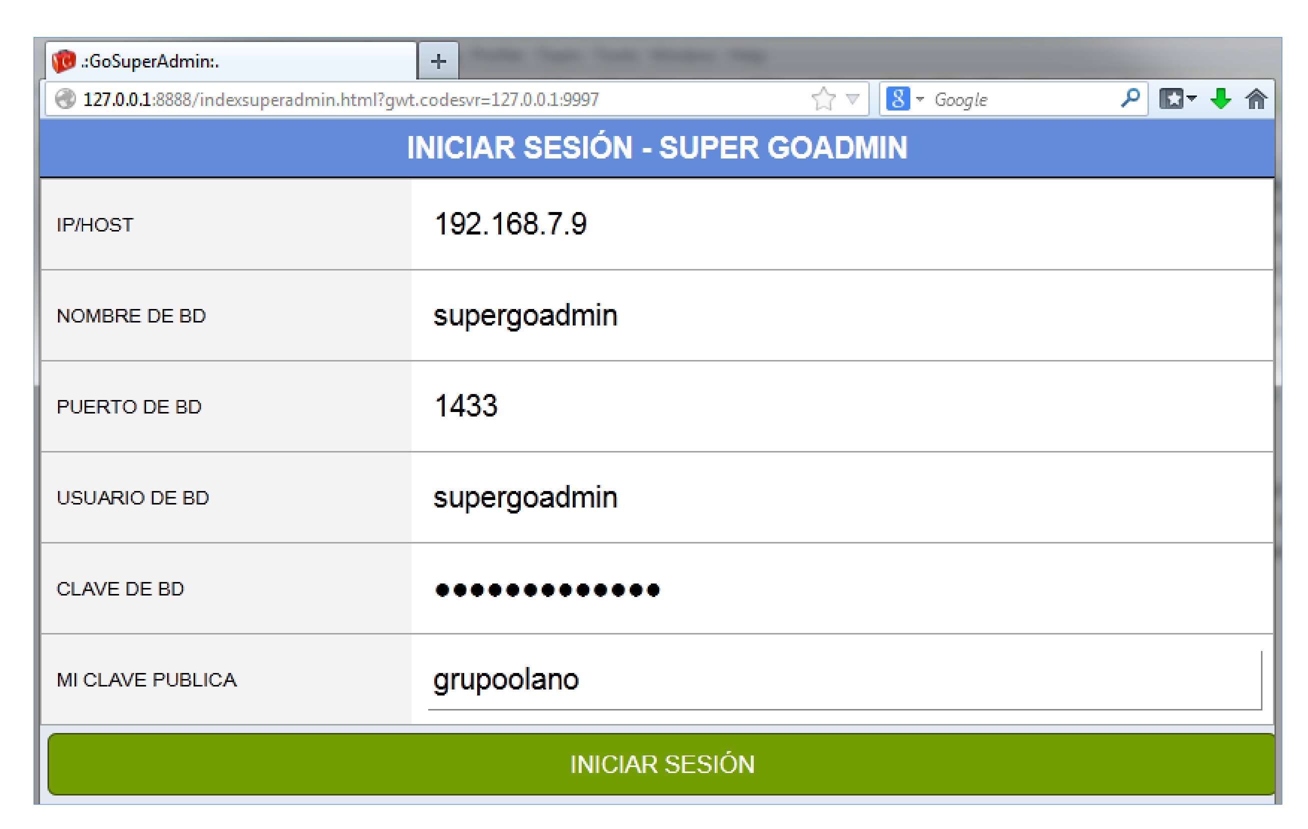


Figura 44: Interfaz de inicio de sesión de supergoadmin

En la figura 45 se muestra la interfaz de inicio de sesión de los usuarios tipo goadmin, los cuales verifican los datos lógicos usando la conexión de supergoadmin y luego intercambian estos datos lógicos por datos de conectividad y así establecen o abren una conexión con su almacén de datos.

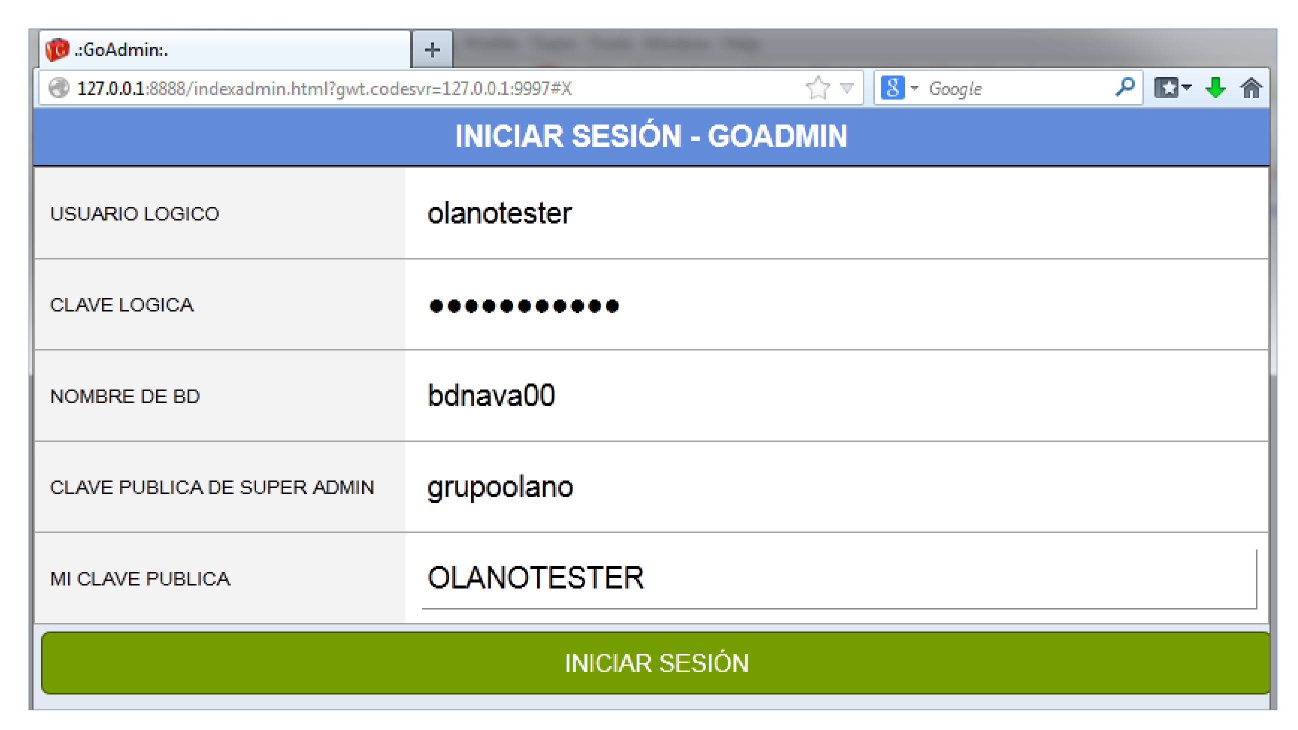


Figura 45: Interfaz de inicio de sesión de usuarios goadmin

En la figura 46 se muestra la interfaz de inicio de sesión de los usuarios gouser, los cuales envían datos lógicos usando la conexión de goadmin, para luego ser intercambiados por datos de conectividad y de esta manera establecer conexión con las base de datos de goadmin.

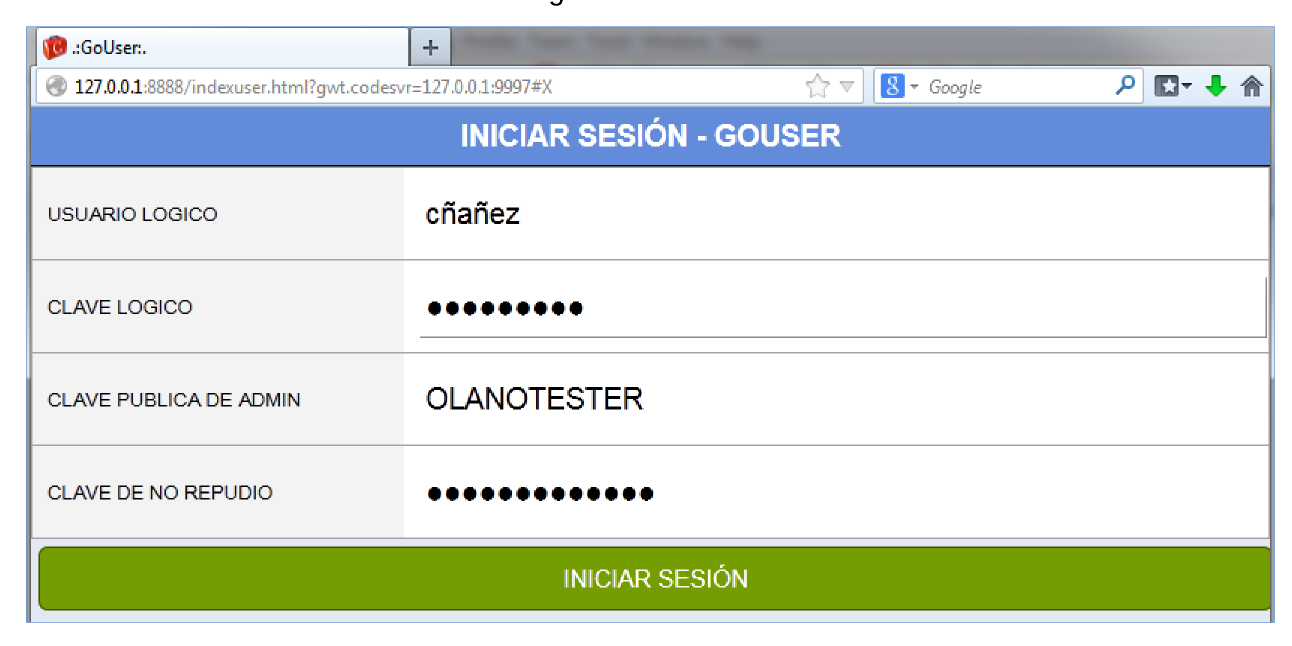


Figura 46: Interfaz de inicio de sesión de usuarios gouser

Como hemos se puede observar el método cumple con el objetivo planteado, el

cual menciona la importancia de salvaguardar los datos de conectividad, así como también servir como un método de autenticación de usuarios que permita tener seguridad, controlo y monitoreo sobre los usuarios del sistema.

2.3.3 Beneficios del método de autenticación.

- Mantener seguros los datos de conectividad al servicio de almacenamiento.
- Permitir autenticar a los usuarios del sistema.
- Administración centralizada de usuarios.
- Seguridad de autenticación basada en 3 niveles.
- Control y monitoreo de usuarios a nivel de sistema y de actividad SQL.
- Balanceo de carga de aplicaciones.
- No repudio al autenticar.

3. Aspecto Administrativo:

3.1 Cronograma de actividades

Actividad			SEMANAS														
			1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Componente 1: Análisis del problema																	
1.1	Analizar la problemática de mantener seguros lo datos de conectividad.																
1.2	Buscar antecedentes que resuelvan el problema.																
1.3	Analizar posibles soluciones al problemas																
Componente 2: Desarrollo de la solución																	
2.1	Desarrollar método de autenticación que permita salvaguardar los datos de conectividad																
Componente 3: Validación de propuesta de solución																	

3.1	Análisis de sistemas para implementar método autenticación en una solución software corporativo.																
3.2	Desarrollo de aplicación web corporativo que implementa método que salvaguarda datos de conectividad																
Componente 4: Contrastación de Resultados																	
4.1	Verificar el logro de objetivos																
4.2	Validar hipotesis																
4	Generar Informe de resultado.																

4. Conclusiones.

- Guardar los datos de conectividad en la memoria RAM de los servidores, permite disminuir las probabilidades de robo de estos datos.
- Al implementar un método de autenticación basada en 3 niveles de seguridad, proporciona características que permiten monitorear y controlar los accesos al sistema.
- El método permite balancear la carga tanto en conexiones a base de datos como a nivel de servidores de aplicaciones, ya que los datos de conectividad son enviados y no almacenados en el código fuente o archivos de texto plano.
- Monitorear la actividad SQL es una las características que proporciona este método de autenticación.
- Controlar el número de conexión o sesiones abierta con los servicios de almacenamiento garantiza el rendimiento de las aplicaciones implementadas con este método.

5. Recomendaciones.

- No colocar los datos de conectividad al servicio de almacenamiento en el código fuente, ni en archivos de texto plano.
- No usar tecnología de conectividad de almacenamiento de datos que obliguen a exponer los datos de conectividad en los clientes, caso de ejemplo ODBC.
- Como desarrolladores, se debe salvaguardar esta información y jamás difundirla.

6. Referencias Bibliográficas.

Bibliografía

(s.f.). Recuperado el 01 de 01 de 2014, de Sitio web de University System of Georgia: http://www.usg.edu/galileo/skills/unit04/primer04_01.phtml

Black, U. D. (1983). *Redes de transmisión de datos y proceso distribuido.* España: Prentice - Hall.

Date, C. J. (2001). *Introducción a los Sistemas de Base de Datos.* Pearson, Prentice Hall.

ISO/IEC. (2005). *ISO/IEC 17799:2005.*

Pons, O., Marín, N., Medina, J. M., Acid, S., & Vila, A. (2009). *Introducción a las Bases de Datos.* España: Thomson Editors.

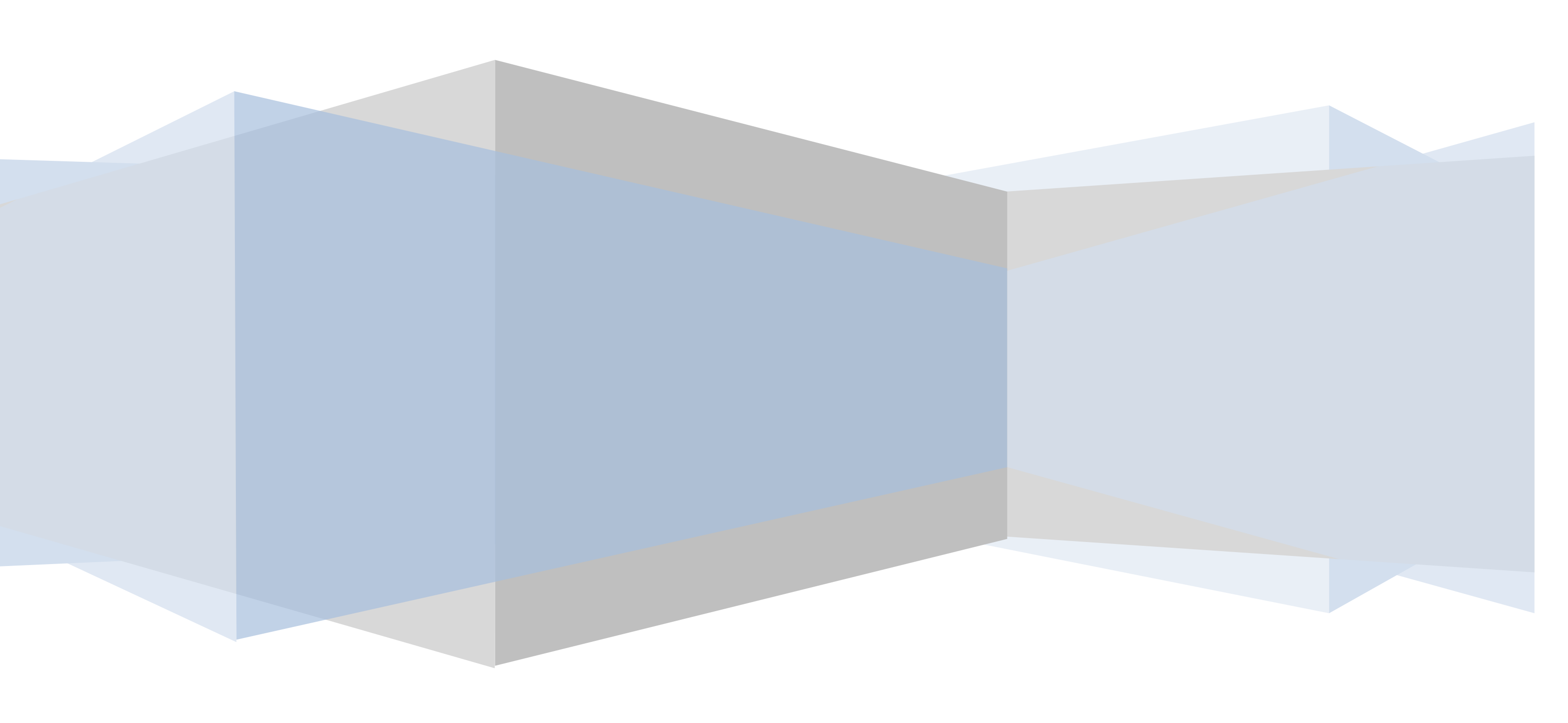
Santamaria, E. (1993). *Electrónica digital y microprocesadores.* Madrid, España: Biblos.

7. Anexo

Anexo D: Documentación del sistema.

MANUAL ADMINISTRATIVO

Aplicando Método de autenticación de clave pública



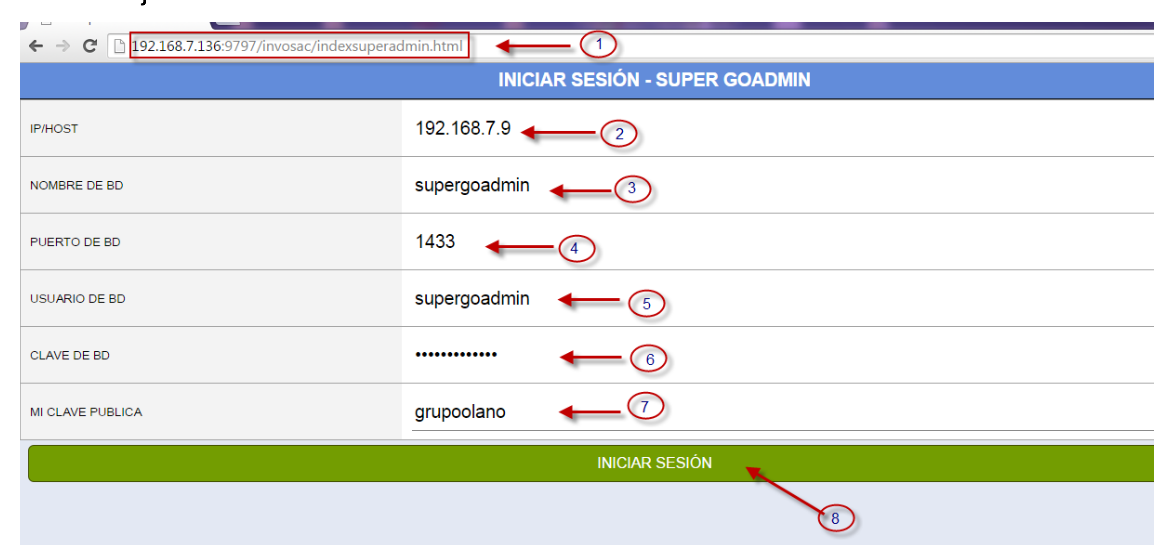
2015

<u>**MANUAL ADMINSTRATIVO**</u>

Este manual solo tendrán acceso los administrativos o personas directas con la aplicación.

Se ha desarrollado 3 niveles de seguridad donde se explicara paso a paso la seguridad de la aplicación.

1. Seguridad ALTA: **indexsuperadmin.html "SUPERGOADMIN"**
   - La URL a seguir es: http://192.168.7.136:9797/invosac/indexsuperadmin.html, en este nivel el formulario se solicita datos para la autenticación de usuario de base de datos, el cual permite establecer la conexión de la aplicación con el sistema manejador de base de datos.



1.-Ingresamos la URL para acceder al nivel más alto de seguridad.

2.- Se ingresa la dirección IP del servidor de la base de datos.

3.- Nombre de la base de datos a la cual establecer conexión.

4.- Puerto del servicio de base de datos.

5.- El nombre de usuario de la base de datos tiene que ser igual al nombre de la base de datos.

6.- La clave de usuario de base de datos, tiene que cumplir un grado de complejidad.

7.- Una palabra clave con significado propio.

8.- Damos clic en Iniciar Sesión para ingresar a la aplicación.

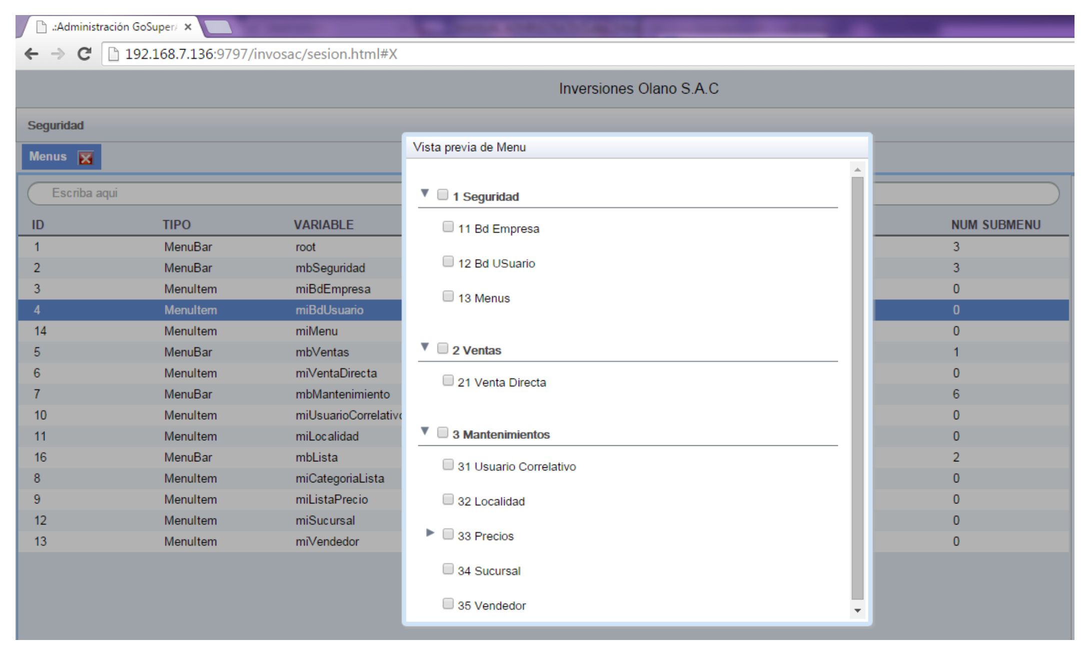
Ingresamos al modulo de **seguridad**



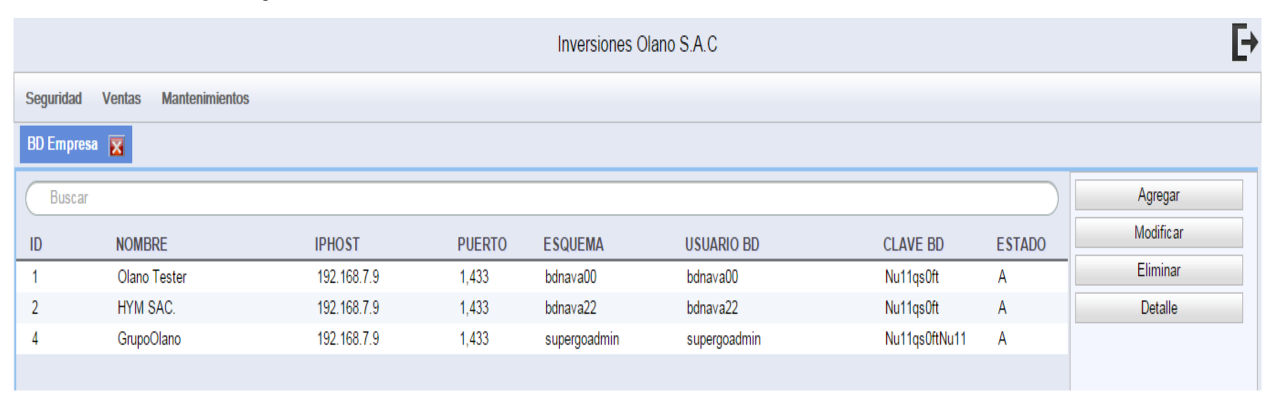
Modulo Menu



Se ingresa a la opción de Menus, aquí se puede visualizar la jerarquía de menus.



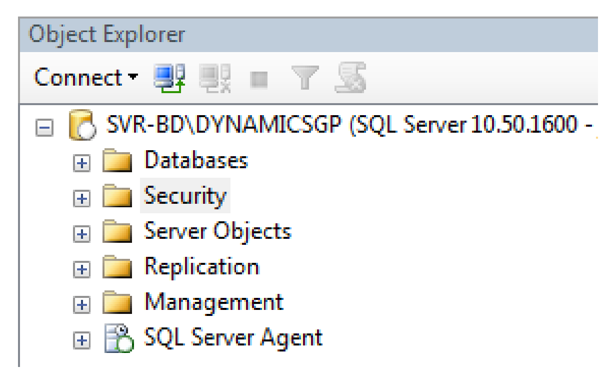
Modulo de Bd Empresas



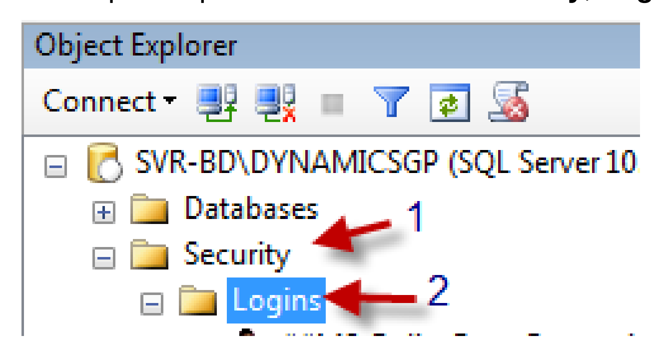
ID	NOMBRE	IPHOST	PUERTO	ESQUEMA	USUARIO BD	CLAVE BD	ESTADO
1	Olano Tester	192.168.7.9	1,433	bdnava00	bdnava00	Nu11qs0ft	A
2	HYM SAC.	192.168.7.9	1,433	bdnava22	bdnava22	Nu11qs0ft	A
4	GrupoOlano	192.168.7.9	1,433	supergoadmin	supergoadmin	Nu11qs0ftNu11	A

Hacer clic en agregar para ingresar una nueva empresa

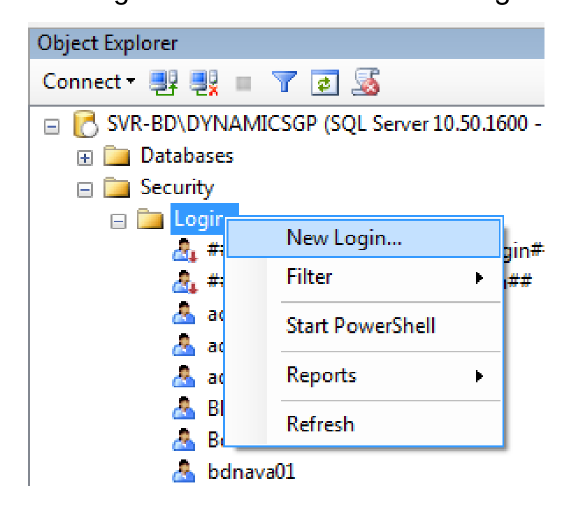
- ❖ Imaginar que vamos a crear una nueva empresa llamada GRUPO OLANO S.A.C su base de datos se llamara Bdnava17.
  - ✓ Antes de crear una empresa en la nueva aplicación tendría que crearla en nava y a partir de allí se clonara la base de datos para la nueva empresa.
  - ✓ Luego iría al Microsoft SQL Server Management Studio.



Como primer paso se hará clic en **Security**, **Logins**



En Logins se hará anti clic New Login… y se ingresara los datos que nos soliciten:



Login - New

Select a page
General
Server Roles
User Mapping
Securables
Status

Script Help

Login name: bdnava17 Search...

1.- Ingresamos el nombre de la base de datos

Windows authentication

SQL Server authentication

2.- Marcamos SQL

Password: ••••••••••

Confirm password: ••••••••••

3. Clave de la base de datos

Specify old password

Old password:

Enforce password policy

Enforce password expiration

User must change password at next login

4. Desactivamos la opción que forzara el cambio de la clave

Mapped to certificate

Mapped to asymmetric key

Map to Credential Add

Mapped Credentials

Credential	Provider

Remove

5.- Seleccionamos la base de datos

Default database: BdNava17

Default language: <default>

Connection

Server:
SVR-BD\DYNAMICSGP

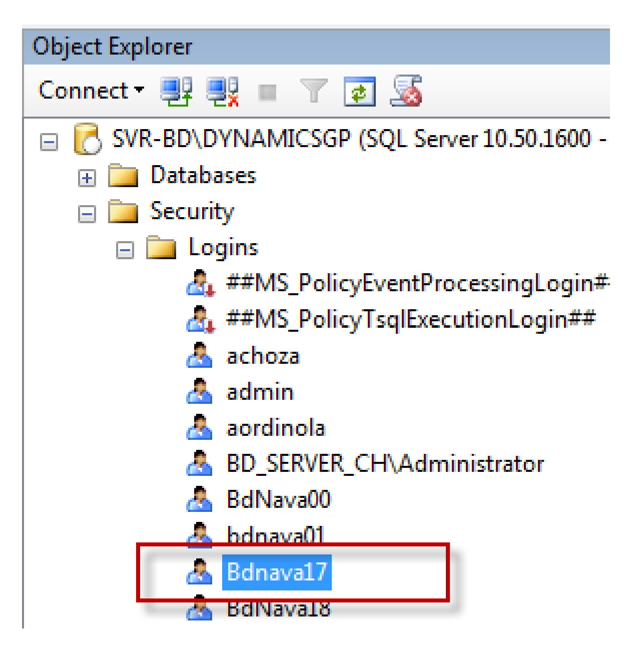
Connection:
jofrantoba

View connection properties

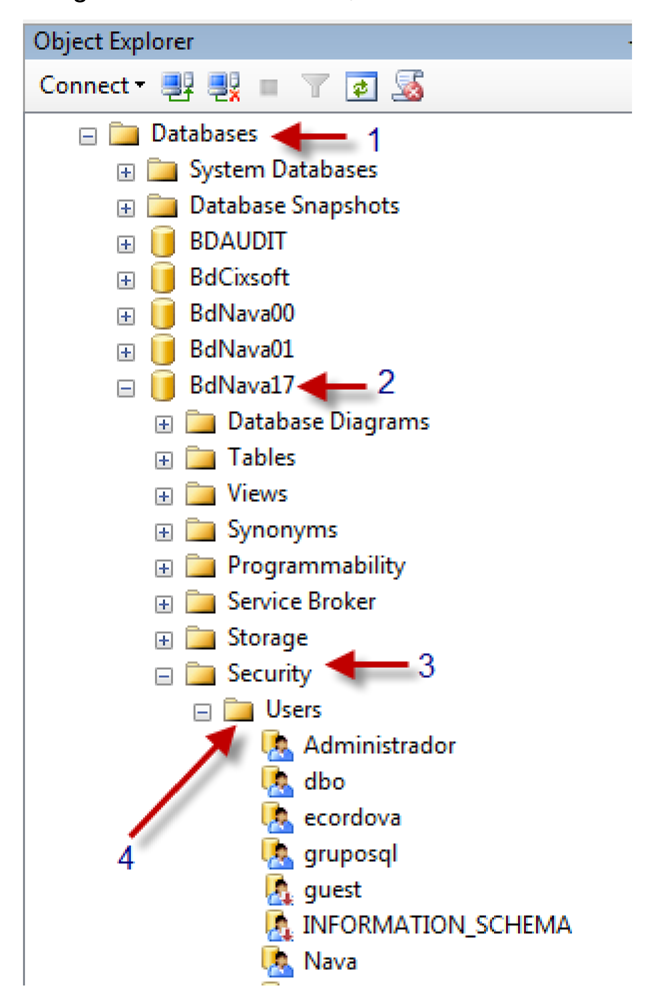
Progress

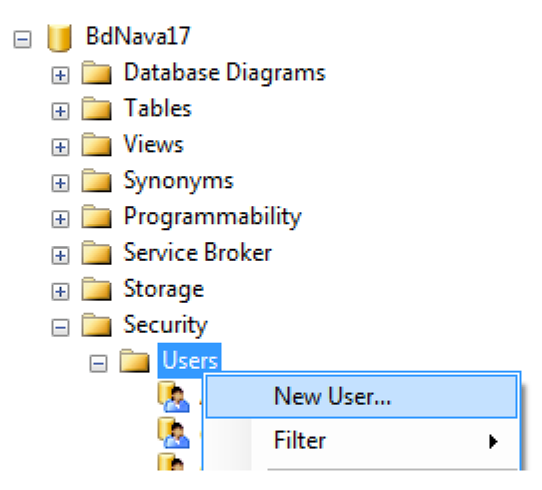
Ready

6.- Aceptamos la creación OK Cancel

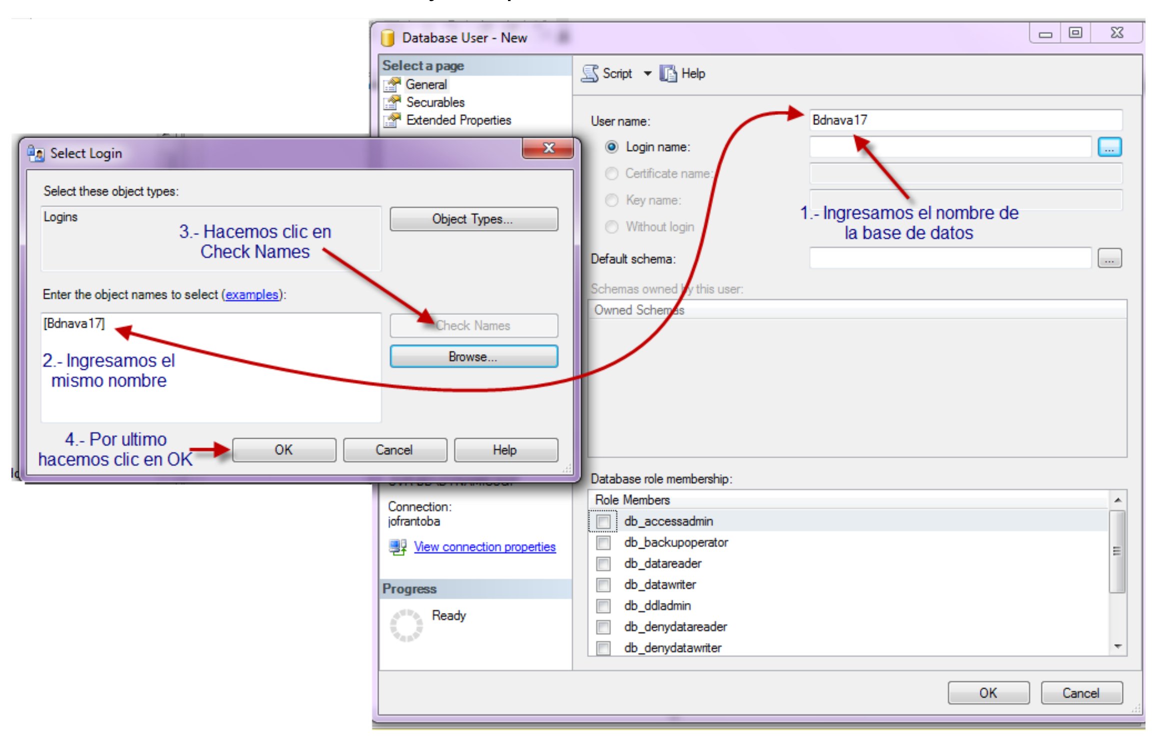


Luego se ira a **Database**, el nombre de la base de datos, **Security.**





En **Users** se dará anti clic New User… y completar los datos solicitados.



Database User - New

Select a page
General
Securables
Extended Properties

Script Help

Nombre de la base de datos

User name: Bdnava17
Login name: Bdnava17
Certificate name:
Key name:
Without login

2.- Comprobar el nombre

Default schema:

Schemas owned by this user:
Owned Schemas

Connection
Server:
SVR-BD\DYNAMICSGP
Connection:
jofrantoba
View connection properties

Progress
Ready

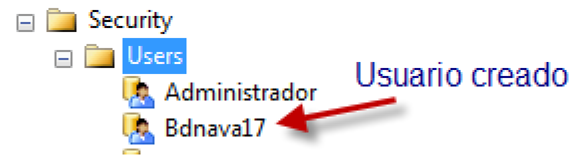
Database role membership:

Role Members
db_datawriter
db_ddladmin
db_denydatareader
db_denydatawriter
db_owner
db_securityadmin

3.- Miembros de la base de datos.

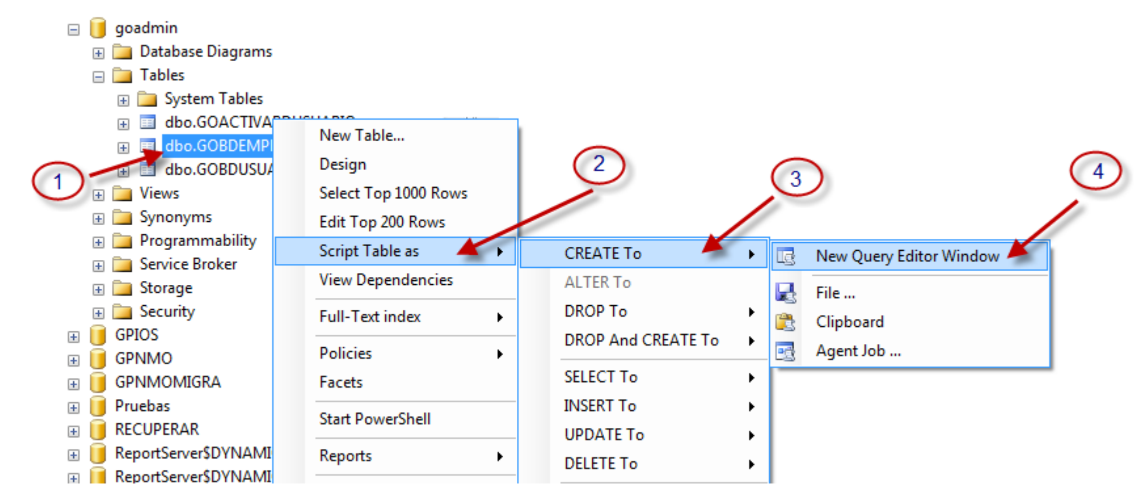
4.- Damos clic en OK

OK Cancel



En la base de datos **Bdnava 17 se** ingresa a tables y se busca **dbo.GOBEMPRESA, dbo.GOBUSUARIO Y dbo.GOACTIVABBDUSUARIO** y si existe se elimina.

Se busca la base de datos **goadmin**, para ir a tables:



- **dbo.GOBEMPRESAS** anti clic Script Table as, CREATE TO, New Query Editor Window.

```
USE [BdNava17]
GO
/****** Object:  Table [dbo].[GOBDEMPRESA]    Script Date: 03/31/2015 15:54:43 ******/
SET ANSI_NULLS ON
GO

SET QUOTED_IDENTIFIER ON
GO

SET ANSI_PADDING ON
GO

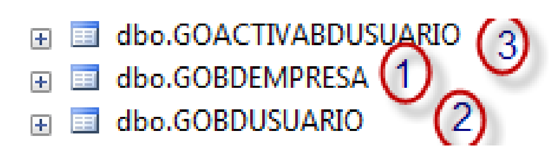
CREATE TABLE [dbo].[GOBDEMPRESA](
    [IDBDEMPRESA] [int] NOT NULL,
    [IPHOST] [varchar](255) NULL,
    [CLAVEPRINCIPAL] [varchar](255) NULL,
    [ESTADOACTIVACION] [varchar](255) NULL,
    [NOMBRE] [varchar](255) NULL,
    [PUERTO] [int] NULL,
    [ESQUEMA] [varchar](255) NULL,
    [USERPRINCIPAL] [varchar](255) NULL,
    [VERSION] [bigint] NOT NULL,
 CONSTRAINT [GOBDEMPRESA_PK] PRIMARY KEY CLUSTERED
(
    [IDBDEMPRESA] ASC
)WITH (PAD_INDEX  = OFF, STATISTICS_NORECOMPUTE  = OFF, IGNORE_DUP_KEY = OFF, ALLOW_ROW_LOCKS  = ON, ALLOW_PAGE_LOCKS
) ON [PRIMARY]

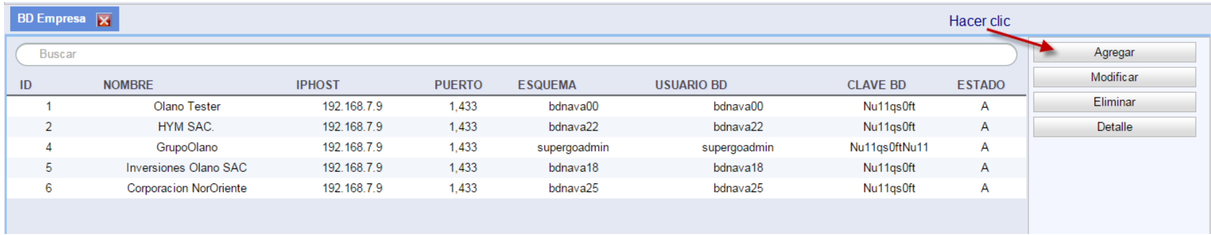
GO

SET ANSI_PADDING OFF
GO
```

Colocamos el nombre de la base de datos

Lo mismo se hace para las tablas **dbo.GOBUSUARIO Y dbo.GOACTIVABBDUSUARIO**

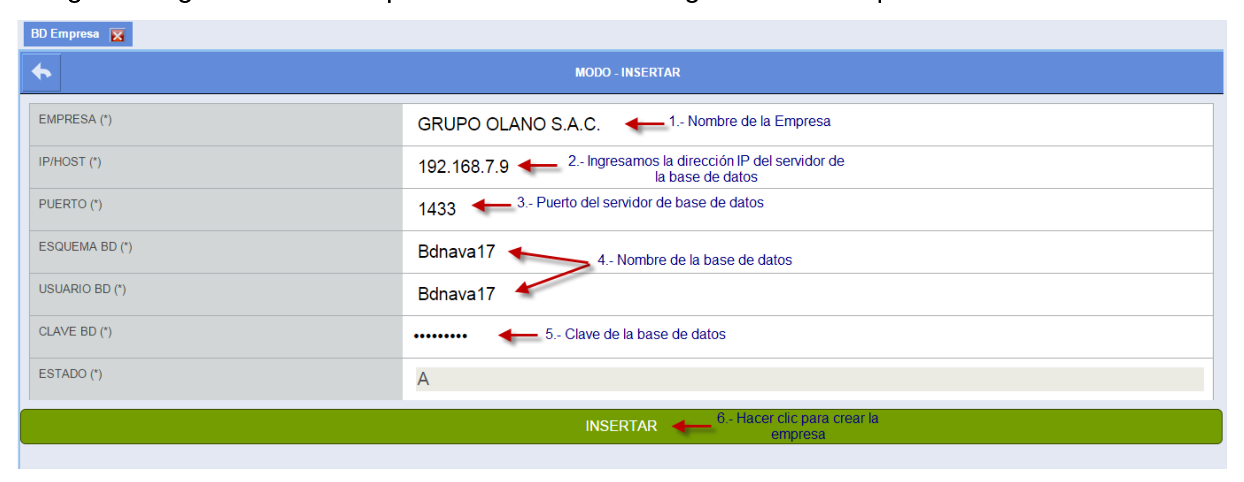


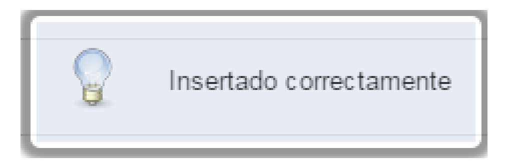


ID	NOMBRE	IPHOST	PUERTO	ESQUEMA	USUARIO BD	CLAVE BD	ESTADO
1	Olano Tester	192.168.7.9	1,433	bdnava00	bdnava00	Nu11qs0ft	A
2	HYM SAC.	192.168.7.9	1,433	bdnava22	bdnava22	Nu11qs0ft	A
4	GrupoOlano	192.168.7.9	1,433	supergoadmin	supergoadmin	Nu11qs0ftNu11	A
5	Inversiones Olano SAC	192.168.7.9	1,433	bdnava18	bdnava18	Nu11qs0ft	A
6	Corporacion NorOriente	192.168.7.9	1,433	bdnava25	bdnava25	Nu11qs0ft	A

Luego se ira a la aplicación y se dara clic en Agregar.

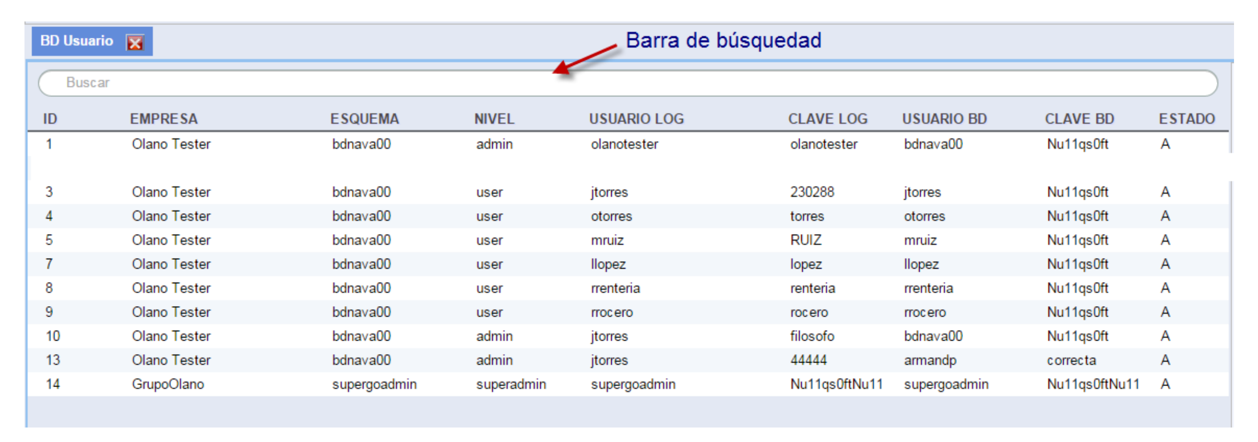
Luego se ingresara los campos solicitados en el registro de la empresa.







ID	NOMBRE	IPHOST	PUERTO	ESQUEMA	USUARIO BD	CLAVE BD	ESTADO
1	Olano Tester	192.168.7.9	1,433	bdnava00	bdnava00	Nu11qs0ft	A
2	HYM SAC.	192.168.7.9	1,433	bdnava22	bdnava22	Nu11qs0ft	A
4	GrupoOlano	192.168.7.9	1,433	supergoadmin	supergoadmin	Nu11qs0ftNu11	A
5	Inversiones Olano SAC	192.168.7.9	1,433	bdnava18	bdnava18	Nu11qs0ft	A
6	Corporacion NorOriente	192.168.7.9	1,433	bdnava25	bdnava25	Nu11qs0ft	A
7	GRUPO OLANO S.A.C.	192.168.7.9	1,433	Bdnava17	Bdnava17	Nu11qs0ft	A

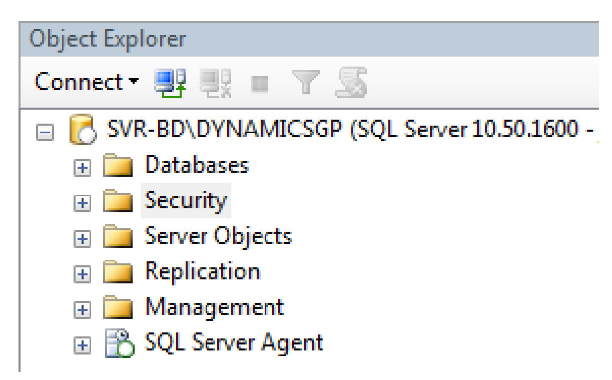


ID	EMPRESA	ESQUEMA	NIVEL	USUARIO LOG	CLAVE LOG	USUARIO BD	CLAVE BD	ESTADO
1	Olano Tester	bdnava00	admin	olanotester	olanotester	bdnava00	Nu11qs0ft	A
3	Olano Tester	bdnava00	user	jtorres	230288	jtorres	Nu11qs0ft	A
4	Olano Tester	bdnava00	user	otorres	torres	otorres	Nu11qs0ft	A
5	Olano Tester	bdnava00	user	mruiz	RUIZ	mruiz	Nu11qs0ft	A
7	Olano Tester	bdnava00	user	llopez	lopez	llopez	Nu11qs0ft	A
8	Olano Tester	bdnava00	user	rrenteria	renteria	rrenteria	Nu11qs0ft	A
9	Olano Tester	bdnava00	user	rrocero	rocero	rrocero	Nu11qs0ft	A
10	Olano Tester	bdnava00	admin	jtorres	filosofo	bdnava00	Nu11qs0ft	A
13	Olano Tester	bdnava00	admin	jtorres	44444	armandp	correcta	A
14	GrupoOlano	supergoadmin	superadmin	supergoadmin	Nu11qs0ftNu11	supergoadmin	Nu11qs0ftNu11	A

Modulo de Bd Usuario

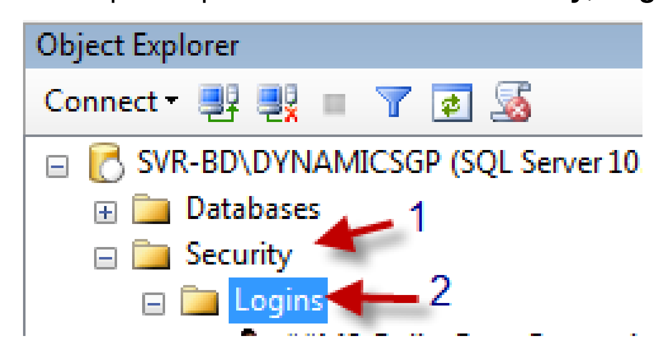
Hacer clic en agregar para ingresar al registro del usuario.

- Imaginar que se va a crear a un usuario Rosa Rodriguez utilizando la base de datos bdnava17
  - ✓ Antes de crear a un usuario en la nueva aplicación el usuario tiene que estar creado en el Nava.

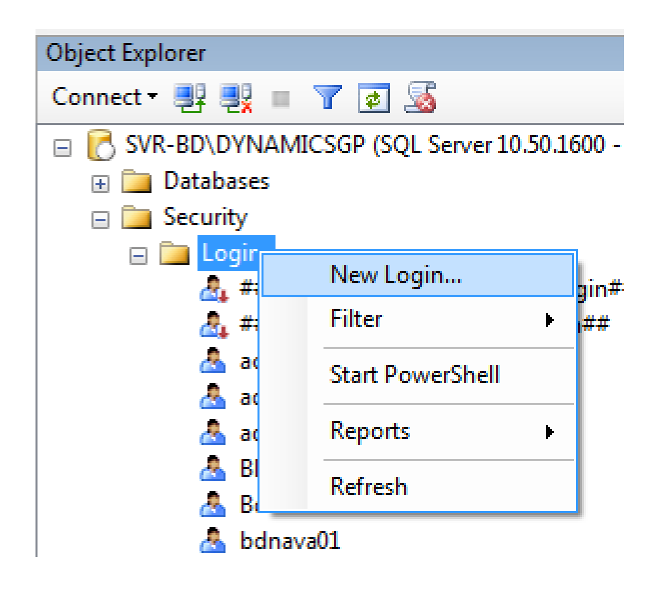


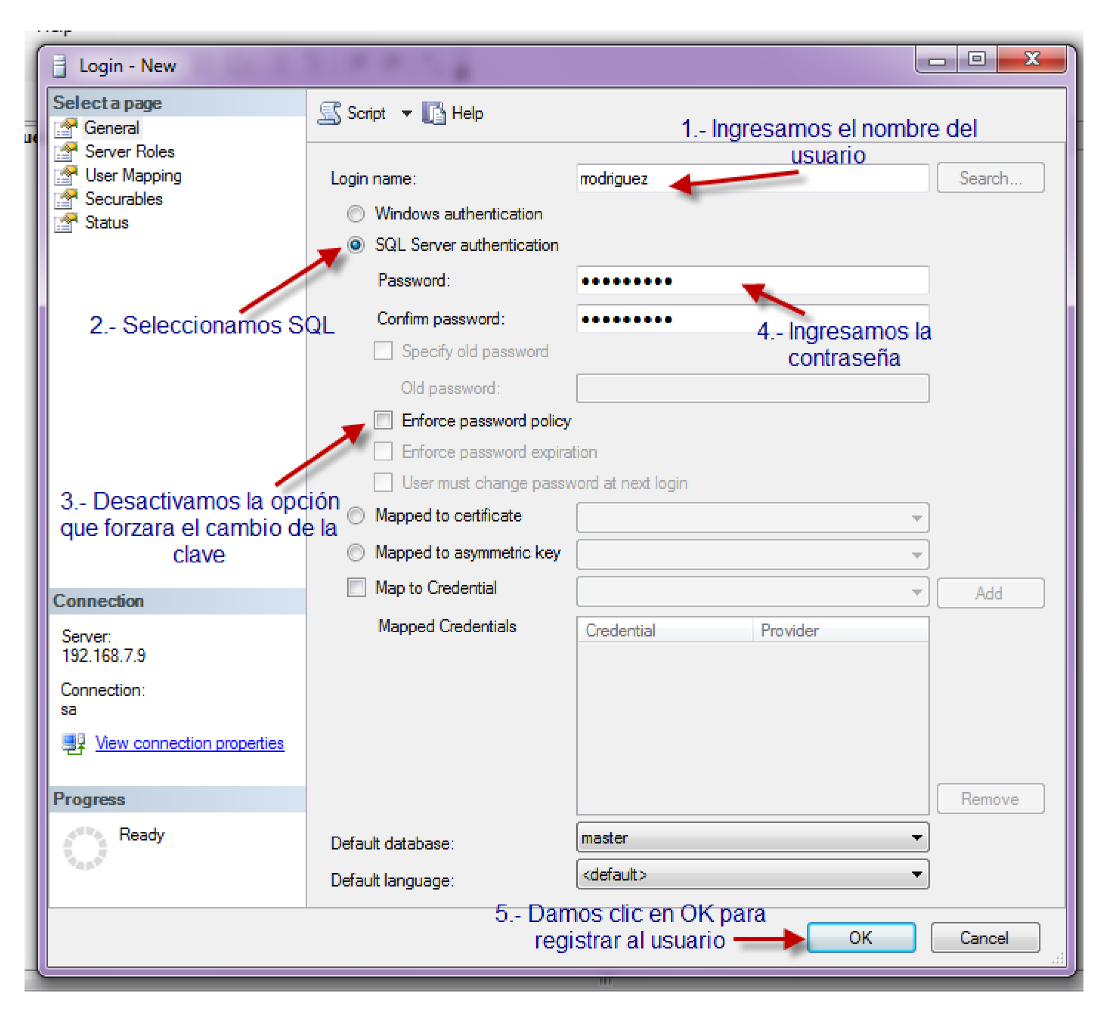
  - ✓ Luego se ira al Microsoft SQL Server Management Studio.

Como primer paso se hará clic en **Security**, **Logins**

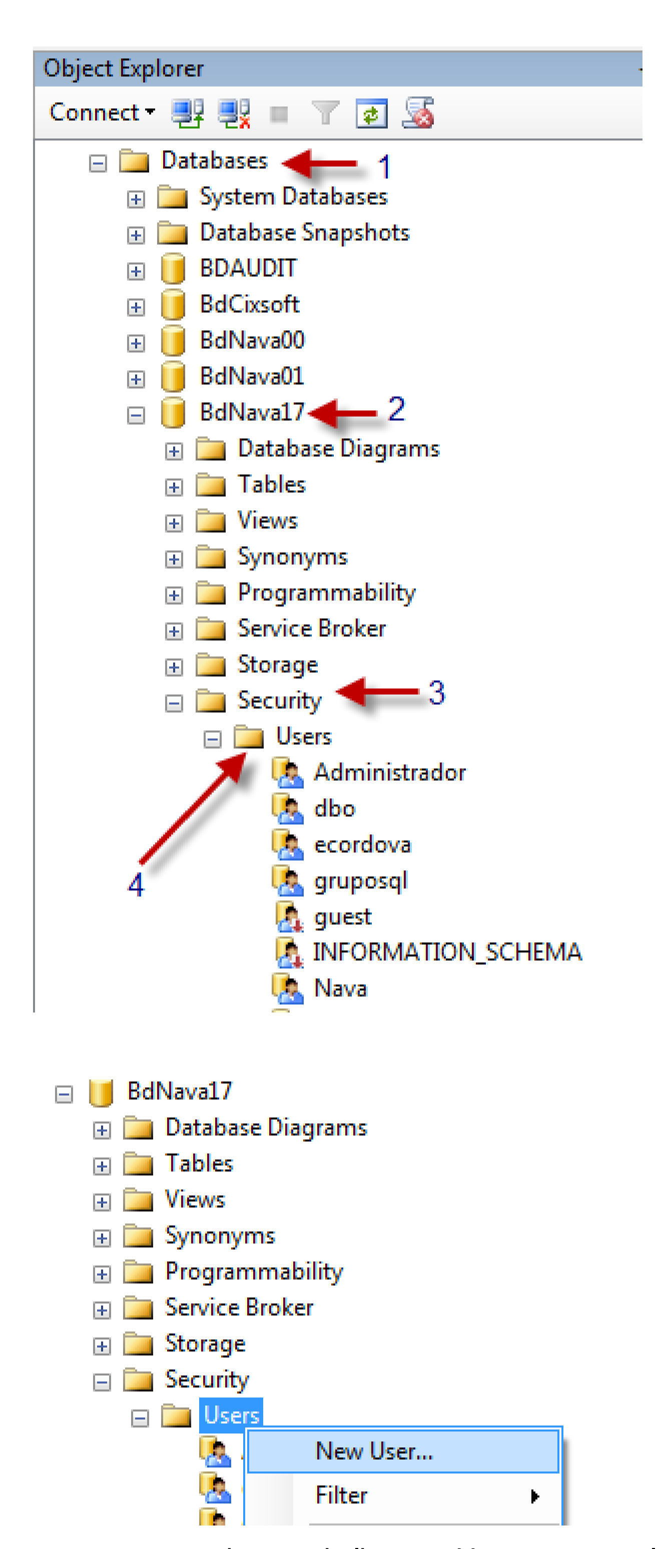


En Logins se hace anti clic New Login… se ingresa los datos que se soliciten:

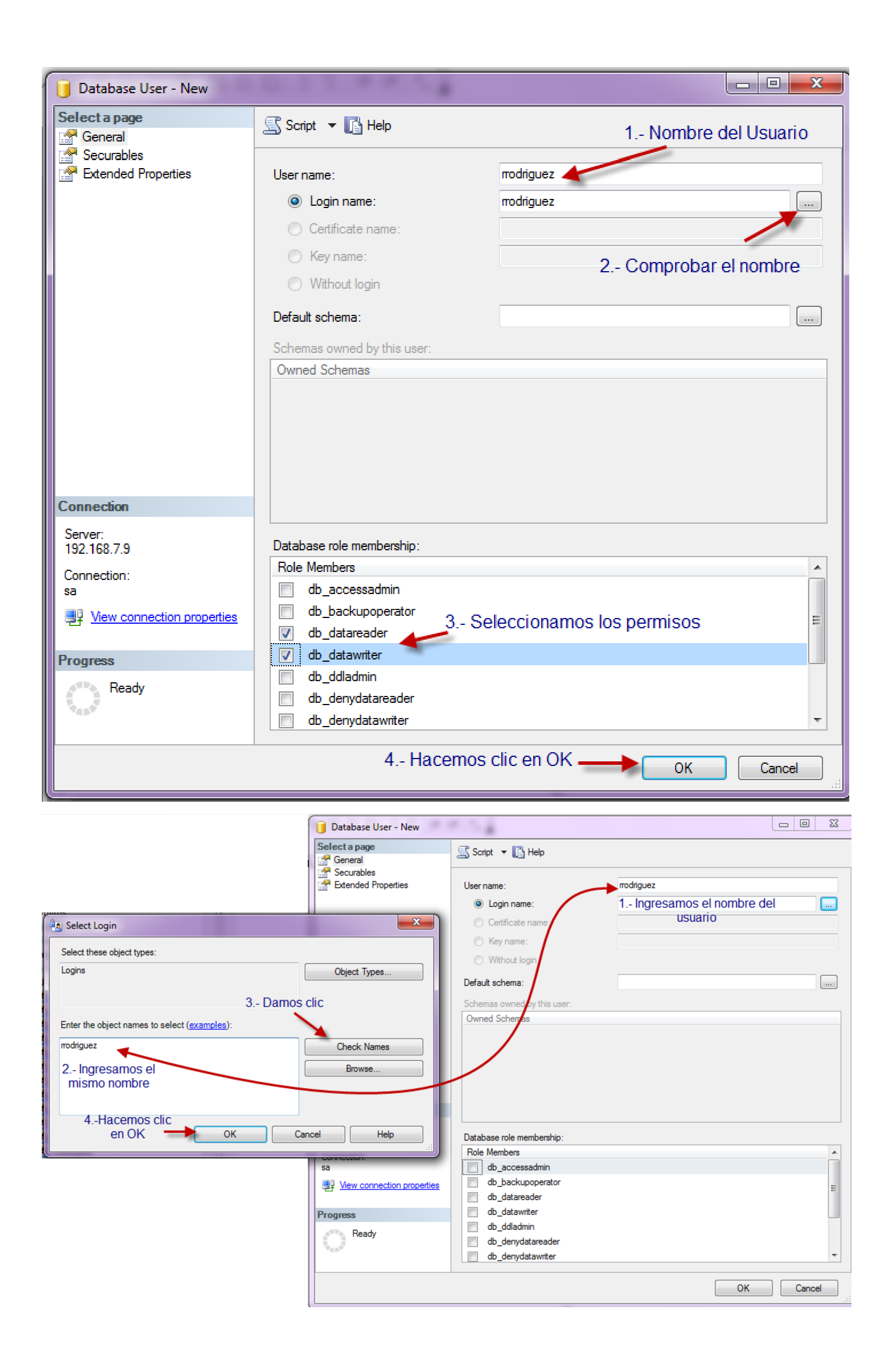


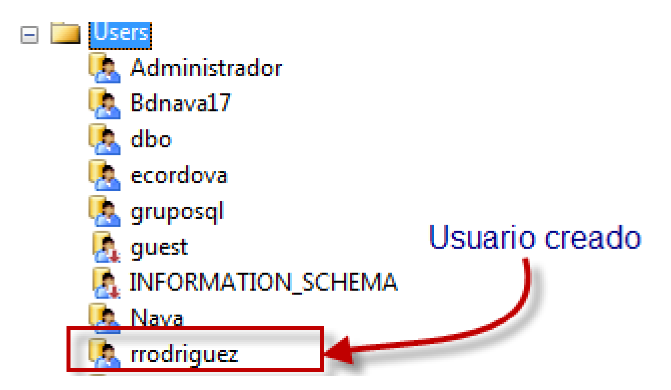


Luego se ira a **Database**, el nombre de la base de datos, **Security.**

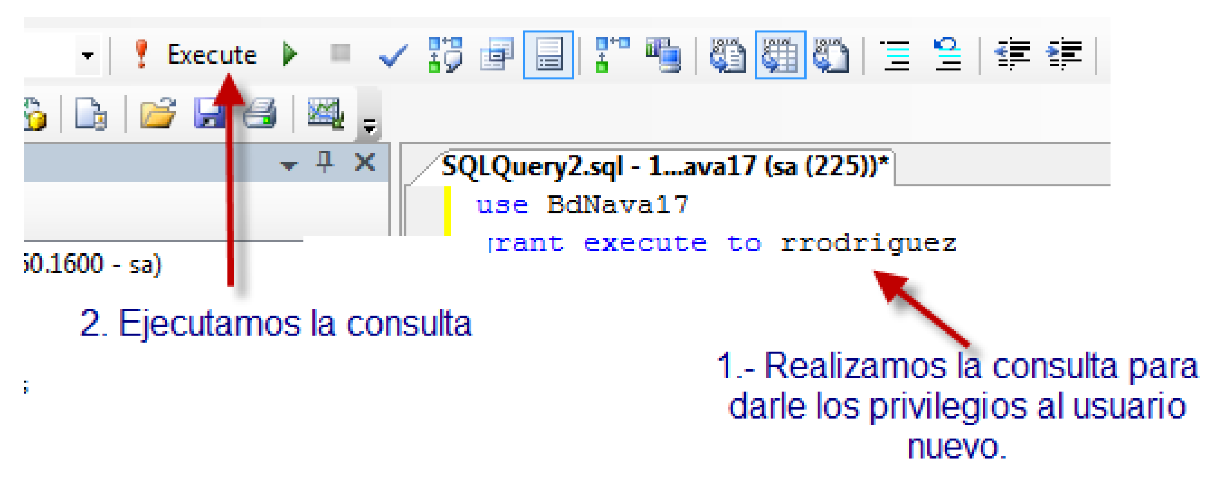


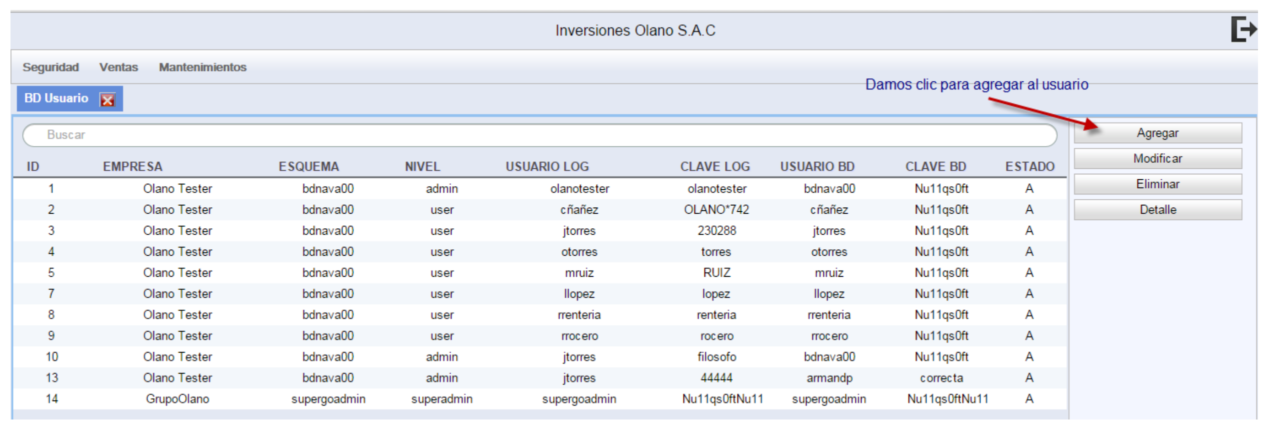
En **Users** **se** dara anti clic New User… y completar los datos solicitados.


Database User - New
Select a page
General
Securables
Extended Properties
Script
Help
1.- Nombre del Usuario
User name:
rrodriguez
Login name:
rrodriguez
Certificate name:
Key name:
Without login
2.- Comprobar el nombre
Default schema:
Schemas owned by this user:
Owned Schemas
Connection
Server:
192.168.7.9
Connection:
sa
View connection properties
Database role membership:
Role Members
db_accessadmin
db_backupoperator
db_datareader
db_datawriter
db_ddladmin
db_denydatareader
db_denydatawriter
3.- Seleccionamos los permisos
Progress
Ready
4.- Hacemos clic en OK
OK
Cancel
Select Login
Select these object types:
Logins
Object Types...
3.- Damos clic
Enter the object names to select (examples):
rrodriguez
Check Names
Browse...
2.- Ingresamos el mismo nombre
4.-Hacemos clic en OK
OK
Cancel
Help
1.- Ingresamos el nombre del usuario



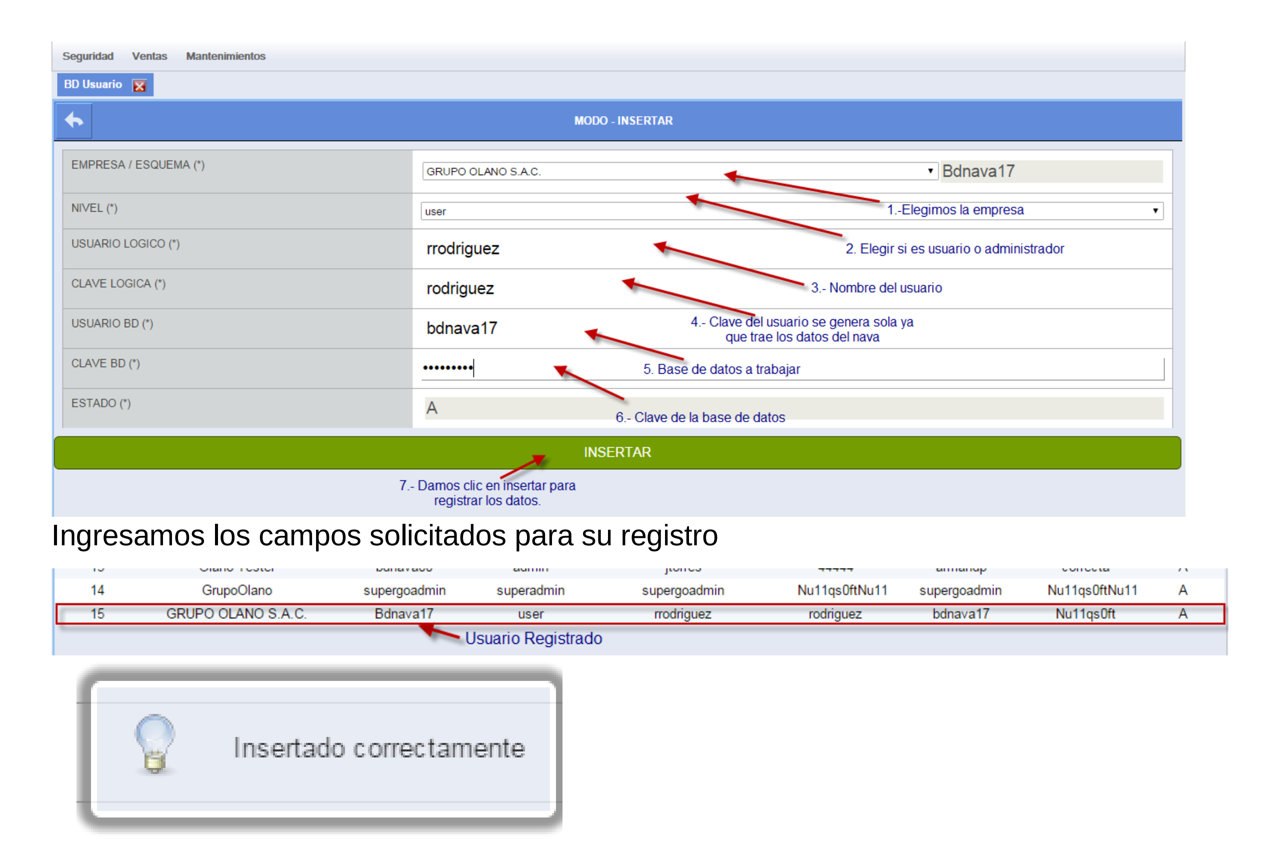
Se ira a la base de datos que se va a trabajar y se ejecutara la siguiente consulta.





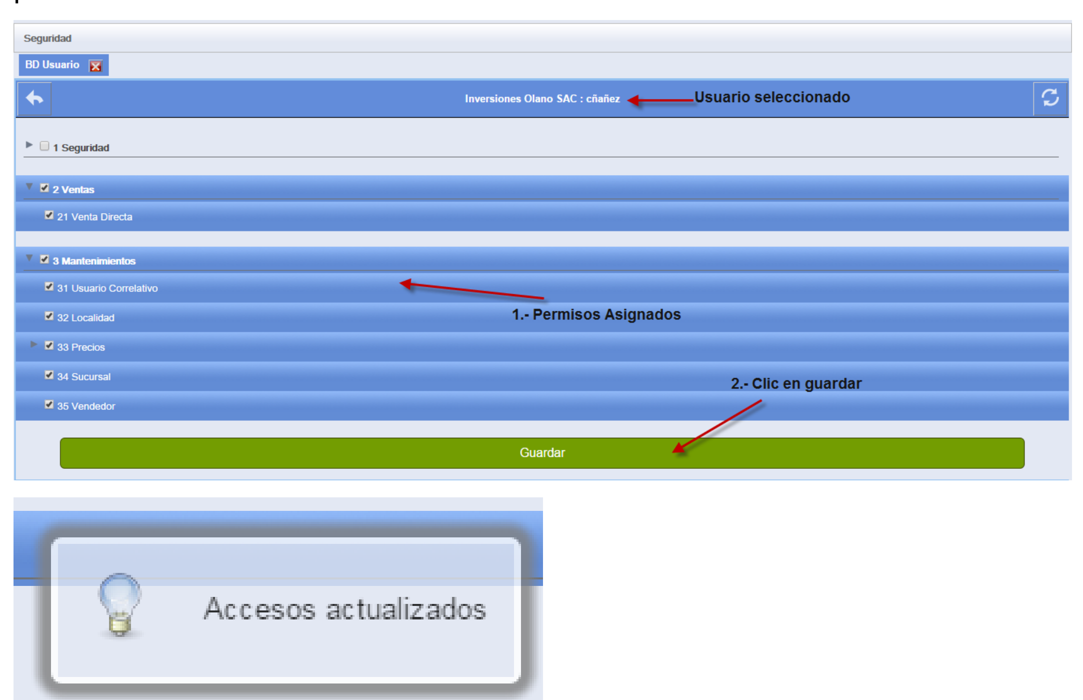
ID	EMPRESA	ESQUEMA	NIVEL	USUARIO LOG	CLAVE LOG	USUARIO BD	CLAVE BD	ESTADO
1	Olano Tester	bdnava00	admin	olanotester	olanotester	bdnava00	Nu11qs0ft	A
2	Olano Tester	bdnava00	user	cñañez	OLANO*742	cñañez	Nu11qs0ft	A
3	Olano Tester	bdnava00	user	jtorres	230288	jtorres	Nu11qs0ft	A
4	Olano Tester	bdnava00	user	otorres	torres	otorres	Nu11qs0ft	A
5	Olano Tester	bdnava00	user	mruiz	RUIZ	mruiz	Nu11qs0ft	A
7	Olano Tester	bdnava00	user	llopez	lopez	llopez	Nu11qs0ft	A
8	Olano Tester	bdnava00	user	rrenteria	renteria	rrenteria	Nu11qs0ft	A
9	Olano Tester	bdnava00	user	rrocero	rocero	rrocero	Nu11qs0ft	A
10	Olano Tester	bdnava00	admin	jtorres	filosofo	bdnava00	Nu11qs0ft	A
13	Olano Tester	bdnava00	admin	jtorres	44444	armandp	correcta	A
14	GrupoOlano	supergoadmin	superadmin	supergoadmin	Nu11qs0ftNu11	supergoadmin	Nu11qs0ftNu11	A

Luego se ira a la aplicación donde se dará clic en **Agregar**



Ingresamos los campos solicitados para su registro

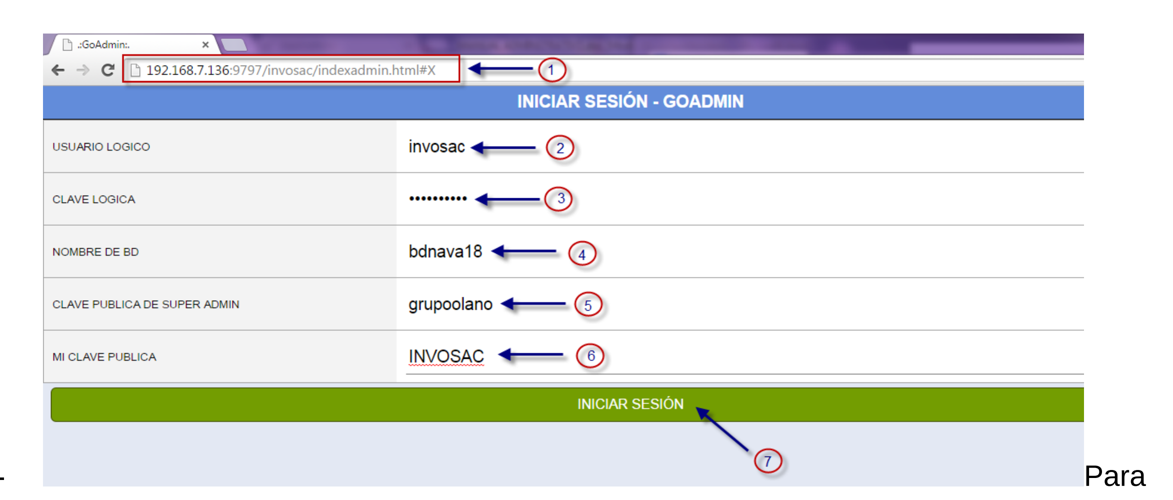
Para asignarle permisos al usuario registrado se ira a la opción **Ver Menu** y se elegirá los permisos.



2. Seguridad Media: **indexadmin.html "GO ADMIN"**

- La URL a seguir es: http://192.168.7.136:9797/invosac/indexadmin.html#X, en este nivel se muestra los datos personales del usuario, permitiendo ocultar los datos de conexión a base de datos al usuario GO ADMIN (seguridad más alta).

- Se tendrá en cuenta que la ***clave pública de super admin*** es la clave general la seguridad más alta en este caso **grupoolano.**



- Para que el superadmin autentique al admin es necesario que cuente con un usuario lógico.

1.- Se ingresa la URL para acceder al nivel medio de seguridad.

2.- SeiIngresa el usuario lógico "**invosac**".

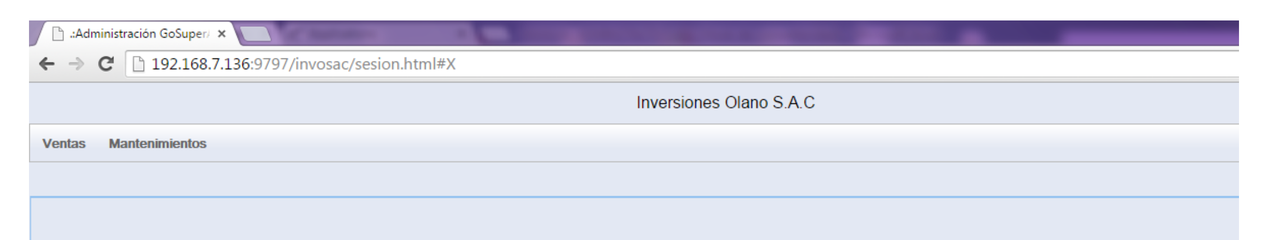
3.- Se ingresa la clave.

4.- Nombre de la base de datos que se va a utilizar.

5.- Se ingresa la clave pública quese coloca en el nivel seguridad alta "**grupoolano**"

6.- En la clave pública se ingresa cualquier palabra.

7.- Se da clic en Iniciar Sesión para ingresar a la aplicación.

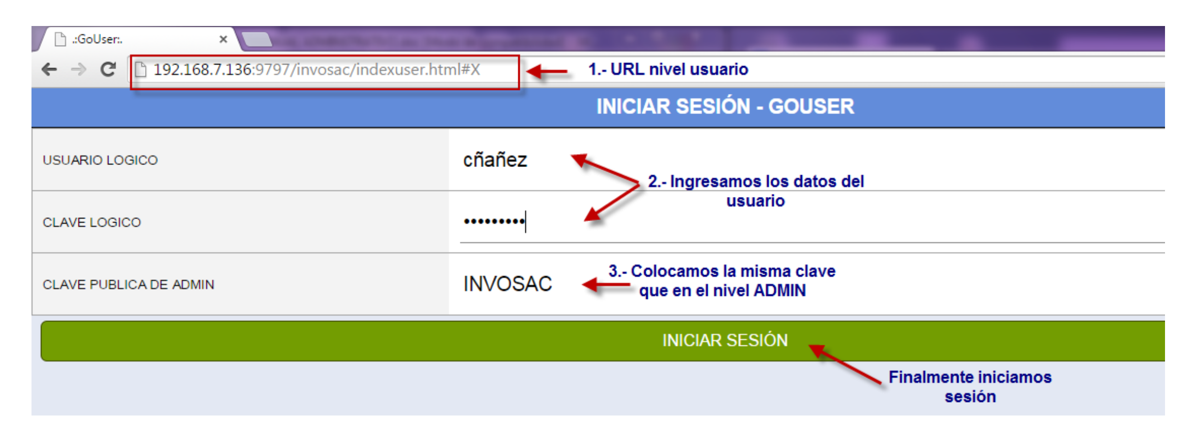


Aquí ya se autentifica al GO ADMIN, esto permite autenticar a los usuarios del nivel GO USER,

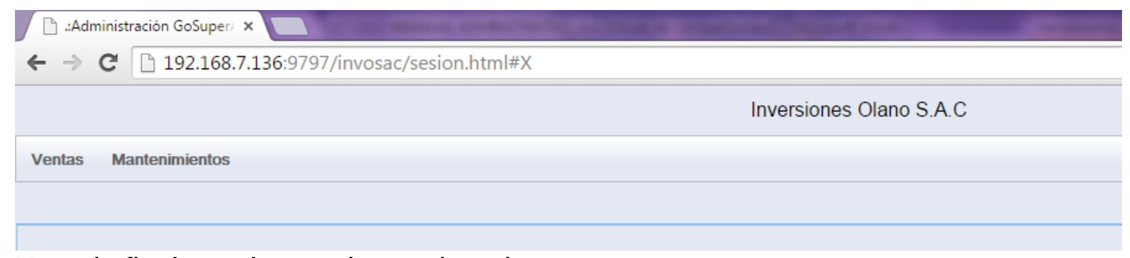
desde aquí también se puede dar los permisos a los usuarios y al mismo, pero no se reflejara los cambios en el nivel de seguridad más alta.

3. Seguridad Usuario: **indexuser.html "GO USER"**

- La URL a seguir es: http://192.168.7.136:9797/invosac/indexuser.html#X, aquí se permite que los usuarios finales del sistema se autentiquen este nivel será el GO USER.



- Es necesario que los usuarios cuenten con un usuario lógico y una clave relacionado con el usuario de la base de datos, la cual es comparada con los niveles anteriores de la autenticación GO ADMIN y SUPER GO ADMIN así se permitirá estar seguros de que la autenticación del usuario es segura.



Usuario final, con lo permisos asignados.